Sequencing and Routing in a Large Warehouse with High Degree of Product Rotation*

Giacomo Lanza Mauro Passacantando Maria Grazia Scutellà

Dipartimento di Informatica, University of Pisa
Largo Bruno Pontecorvo, 3, 56127 Pisa, Italy,
giacomo.lanza@di.unipi.it, mauro.passacantando@unipi.it,
maria.grazia.scutella@unipi.it

Abstract

The paper deals with a sequencing and routing problem originated by a real-world application context. The problem consists in defining the best sequence of locations to visit within a warehouse for the storage and/or retrieval of a given set of items during a specified time horizon, where the storage/retrieval location of an item is given. Picking and put-away of items are simultaneously addressed, by also considering some specific requirements given by the layout design and operating policies which are typical in the kind of warehouses under study. Specifically, the considered sequencing policy prescribes that storage locations must be replenished or emptied one at a time by following a specified order of precedence. Moreover, two fleet of vehicles are used to perform retrieving and storing operations, whose routing is restricted to disjoint areas of the warehouse. We model the problem as a constrained multicommodity flow problem on a space-time network, and we propose two Mixed-Integer Linear Programming formulations, whose primary goal is to minimize the time traveled by the vehicles during the time horizon. Since large-size realistic instances are hardly solvable within the time limit commonly imposed in the considered application context, a matheuristic approach based on a time horizon decomposition is proposed. Finally, we provide an extensive experimental analysis aiming at identifying suitable parameter settings for the proposed approach, and testing the matheuristic on particularly hard realistic scenarios. The computational experiments show the efficacy and the efficiency of the proposed approach.

Keywords. Logistics; Inventory; Warehouse management; Pick-up and Put-away routing; Matheuristic.

1 Introduction

In any supply chain, warehouses play a critical role. Warehousing concerns receiving, storing, order picking, and shipping of goods. In particular, order picking – the process of retrieving items from their storage locations in response to customer orders (Masae et al., 2020) – is often referred to as the most labor- and time-consuming internal logistics process. The large majority of all order picking systems is operated according to the picker-to-parts principle (especially in Western Europe according to van Gils et al., 2018), i.e., pickers walk or drive through the warehouse to retrieve products. The largest portion of an order picker's time is spent on

*This paper has been published in Flexible Services and Manufacturing Journal, vol. 35, pp. 1206-1255, DOI: 10.1007/s10696-022-09463-w, see: https://link.springer.com/article/10.1007/s10696-022-09463-w.

travelling between locations. The cost of these operations is estimated to be approximately the 50% of the total operating cost of a warehouse (De Koster et al., 2007). As judged as a key point to improve productivity and decrease operational costs, the order picking problem has been widely studied in recent years (van Gils et al., 2018; Masae et al., 2020). However, in many warehouses, pickers frequently face not only the picking, but also the stocking of products. If we also consider the storage, the careful and efficient organization of workers operations becomes even greater. Nevertheless, this integrated problem has received less attention (de Brito and De Koster, 2004; Ballestín et al., 2013).

The performance of order picking and storing operations heavily depends on the locations where the goods to retrieve or store are situated or have to be situated. The possible locations for a stock keeping unit (SKU) can be broadly established from the type of *storage policy* followed in the warehouse. The most common storage policies are the dedicated storage policy, which prescribes a particular location for each SKU needing storage, the random storage policy, which involves the random assignment of SKUs to any available and eligible location within the storage area, and the class-based policy, which aims at storing groups of products at nearby positions as they are often required simultaneously (Rouwenhorst et al., 2000; De Koster et al., 2007; Gu et al., 2007). Operationally, the exact position in the warehouse is then addressed each time SKUs need to be stored, possibly subject to additional rules depending on the specific application context.

The problem we consider in this paper arises once a set of items needs to be moved towards their chosen storage locations within the warehouse (put-away operations) and a different set of items needs to be retrieved to fulfill customer order requests (picking operations). This problem is known in the literature as Sequencing and Routing Problem (SRP) and has the scope of defining the most efficient sequence of operations to move SKUs within the warehouse and perform order picking and put-away operations. The objective is typically to minimize the total material handling cost or travel efforts (measured either in time or distance traveled by the workers), while respecting some additional and peculiar rules related to the application context (De Koster et al., 2007).

The work has been motivated by the study of a real application involving a large production site of an Italian company located in Tuscany. It is composed of a production area and a large unit-load warehouse. Its modernization is the goal of a big research project funded by Regione Toscana and it includes the resolution of a SRP for order picking and put-away operations via Operations Research techniques. The involved warehouse is larger than 10,000 m^2, has a rectangular internal layout composed of narrow storage aisles and wide cross aisles and is comprised almost entirely of storage areas. Thus, the distances traveled to perform operations are very large. SKUs are homogeneously stocked into storage locations, i.e., different types of products cannot share the same location, which is accessible only frontally from storage aisles. The warehouse relies on a random storage policy and it is characterized by a high product rotation index, i.e., more than 1,000 SKUs are moved per day. A pick-and-sort policy is also applied. Retrieved items are collected in a specific area of the warehouse (a collection area), where SKUs are sorted to establish order integrity before shipping.

The specific requirements and the warehouse design of our industrial partner allowed us to deepen some rarely discussed aspects in the literature on SRPs. Firstly, due to the particular kind of products stocked in the warehouse (tissue products for sanitary and domestic use), a strictly first-in first-out (FIFO) *sequencing policy* needs to be considered for the picking operations, prescribing that picking operations per product type must be performed by considering the time of permanence of SKUs in the warehouse. That is, oldest SKUs have to be retrieved and shipped first. This policy is largely adopted in (but not restricted to) the tissue sector to avoid the deterioration and perishability of goods (e.g., in the food sector, where items closer

to their expiration date are first retrieved). As pointed out in Masae et al. (2020), the routing of pickers subject to precedence-constraints (PC) has only attracted little attention so far, without carefully considering the importance these constraints have in practice. In our context, this also implies some specific rules to consider during the sequencing of put-away operations. Specifically, the storage locations assigned to a product type have to be filled one at a time, to ease follow the FIFO policy later on during retrievals.

Moreover, two types of multi-shuttle fleets of vehicles are available to support workers in warehousing operations. However, the two types of vehicles may only travel on disjoint parts of the production site. For stock replenishment, this implies the design of a two-echelon route to move SKUs to their respective assigned storage locations, mandatorily passing through intermediate capacitated interchange points where SKUs are transferred from one vehicle type to the other one. Multiple interchange points are available within the warehouse, that is there are many alternatives where to finish the first-echelon and where to begin the second-echelon routing. Such a routing scheme is not often discussed in the literature even though it may be frequently encountered in several realistic contexts such as large end-of-line warehouses, automated storage/retrieval or human-robots shared warehouse systems. In addition, it is applied in many of the warehouses operated by our industrial partner. Some contributions discussing SRPs with different skilled fleets of vehicles have been recently considered (Ballestín et al., 2013, 2020). However, their focus is to assign (storage or retrieval) operations to the suitable skilled vehicles. To the best of the authors' knowledge, routing restrictions of vehicles within the warehouse have not been considered so far for picker-to-part warehouse systems. Additional side constraints are also considered.

We formulate the addressed SRP problem in terms of a constrained multicommodity flow problem on an time-space network, and we propose a Mixed-Integer Linear Programming (MILP) model. The SRP may be considered as a variant of the capacitated vehicle routing problem with additional constraints and it is therefore classified as NP-hard (Cuda et al., 2015; Scholz et al., 2016; Masae et al., 2020). Since real instances make the problem very hard to solve, we also propose an alternative problem formulation and a matheuristic approach, based on time decomposition, which is able to solve the problem in reasonable time. Computational experiments on real-world data show the efficiency and efficacy of the proposed approach.

The main contributions of this paper can be summarized as follows: i) we study a new SRP addressing picking and put-away operations with precedence constraints and routing restrictions; ii) we provide two MILP formulations for the addressed SRP; iii) we propose a matheuristic resolution approach able to effectively determine good quality solutions to large-scale problem instances in short time; iv) we present the results of an extensive experimentation underscoring the performance of the proposed matheuristic for real large-scale instances provided by our industrial partner; v) since the MILP formulations cannot be optimally solved on such real large-scale instances, we present the results of additional tests on a set of smaller artificial instances, based however on real data, in order to provide both a comparative evaluation of the results provided by the matheuristic approach, and also some managerial insights on which aspects of the addressed SRP make its resolution particularly complex.

The paper is organised as follows. Section 2 reviews the main results from the literature in the area of SRP. Section 3 describes the SRP addressed in this paper. Section 4 presents the multicommodity flow based MILP formulations for the considered SRP. The matheuristic approach built to tackle the problem is presented in Section 5. Section 6 describes the experimental plan and reports the results of the computational experiments we performed. Finally, Section 7 concludes the paper and identifies some future research directions.

2 Literature Review

The SRP is an important operational problem whose aim is to define the best sequence of locations to visit within a warehouse for storing and/or retrieving a given set of items, where the storage/retrieval location of an item is given (Gu et al., 2007). In picker-to-part warehouses (the vast majority in Western Europe according to van Gils et al., 2018), operations are performed by workers who walk or drive along the aisles transporting items on vehicles, trolleys or carts. Generally, their route starts from and end in a prespecified spot within the warehouse (where they are given the list of the storage/retrieval locations to visit). The SRP normally depends on a number of warehouse-specific features such as the internal layout (e.g., length and number of aisles, presence of cross-aisles, I/O locations), the physical characteristics of the products to move (e.g., type, weight, height, shape), and the specific application context (e.g., storage policy, arrival times of products, shipment due dates, available vehicles types). Usually, the objective of SRPs relates to the optimization of travel efforts or handling times of SKUs.

Surveys on SRPs can be found in van Gils et al. (2018) and Masae et al. (2020). The authors categorized the existing literature with regard to performance measures, modelling methods and combined problems, as well as to type of algorithms (exact, heuristic, and matheuristic) and warehouse internal layout (conventional and non-conventional), respectively. Davarzani and Norrman (2015) and Gong and De Koster (2011), instead, focus on real applications and stochastic approaches, respectively. We refer to De Koster et al. (2007) and Gu et al. (2007) for a more general overview on the operational issues in warehousing problems.

More in detail, the SRP for picking activities only is a well-studied topic in the scientific literature, displaying an increasing trend of interest over the last decade (Masae et al., 2020). The most recent contributions focus on realistic aspects such as particular layout designs (Mowrey and Parikh, 2014; Scholz et al., 2016; Boysen et al., 2017; Weidinger et al., 2019; Briant et al., 2020), congestion issues (Pan and Wu, 2012; Chen et al., 2013, 2016), workers comfort (Grosse et al., 2015) and dynamic modification of list of operations (Lu et al., 2016; De Santis et al., 2018). As opposed, Gómez-Montoya et al. (2020) is the only contribution addressing exclusively a put-away SRP.

Instead of summarizing the vast body of literature on these topics, we refer the interested reader to the recent above-mentioned review papers, focusing here only on those papers addressing some of the peculiarities of the SRP addressed in this work. Specifically, the main features discussed here are: i) the joint sequencing and routing for picking and put-away; ii) the use of peculiar precedence constraints in picking SKUs; iii) the use of heterogeneous equipment and the requirement of routing restrictions within the warehouse.

2.1 Joint storage and retrieval in SRP

The literature dealing with picker-to-part warehouse systems has focused almost exclusively on designing picking routes, making contributions focused on the combination of both picking and put-away much more scarce. A reason is due to the fact that not all picker-to-part warehouse systems are designed or operated under double command operations, i.e., where the storage needs to be planned in combination with the picking process. Double command operations certainly define difficult larger problems to tackle, since they require the simultaneous organization of two different flows of items, often requiring distinct management rules (Gu et al., 2007). Nevertheless, integrated schedule planning of storage/picking operations can provide the opportunity to assign resources more efficiently and have better performances from a practical point of view (Davarzani and Norrman, 2015; Masae et al., 2020).

In other warehousing systems, instead, picking and put-away are more often accounted for together, as for instance in part-to-pickers systems, where items are automatically moved

between storage locations and workers by robotized or semi-robotized storage and retrieval systems, such as stacker cranes or multi-shuttles Gagliardi et al. (2012), or in specific compact-warehousing systems, such as containers in ports or steel slabs in yards Carlo et al. (2014). However, the considered movement restrictions of the equipment make the problem different with respect to ours.

Focusing on picker-to-parts systems, and considering the addressed variants of SRP and the methodologies proposed to solve them, we mention:

- Wruck et al. (2013), proposing multi-objective minimization models for the single-worker case, in both static and dynamic settings;
- Schrotenboer et al. (2017), modeling the single-worker variant of SRP in terms of the traveling salesman problem and solving it through a genetic algorithm; the multiple-worker case is also addressed and solved by modifying single-worker routes;
- Ballestín et al. (2013), modeling the SRP as a project scheduling problem, in both static and dynamic settings.

Moreover, since a crucial aspect that may influence the system performance is the layout of the warehouse, at this regards we mention:

- Pohl et al. (2009a), addressing the three most common rectangular layouts under single-command and dual-command routing protocols;
- Pohl et al. (2009b) and Gue et al. (2012), addressing uncommon layouts such as the Flying-V and Inverted-V aisles design;
- Ballestín et al. (2013) and Ballestín et al. (2020), addressing a chaotic warehouse where SKUs are arranged in parallel aisles composed of multi-level double-depth racks.

2.2 Precedence constraints in SRP

A precedence constraint (PC) imposes that some products must be picked before others due to some restrictions (Matusiak et al., 2014). Restrictions may vary in nature and may be related to weight or fragility issues (e.g., first retrieving heavy items), shape and size of SKUs (e.g., first retrieving big boxes), perishability (e.g., first retrieving those items closer to their expiration date), other product-category specific properties (e.g., to avoid contamination between food and non-food products) or, even, to preferred unloading sequence at customer locations. According to Masae et al. (2020) and van Gils et al. (2018), PCs in picking operations have attracted little attention so far, even though they are encountered very often in realistic contexts.

Focusing on the considered PCs and/or on the methodologies proposed to address them, we mention here:

- Matusiak et al. (2014), considering PCs related to the type of product, and solving a variant of the precedence-constrained travelling salesman problem via a decomposition based heuristic approach;
- Chabot et al. (2017), considering weight and product-category PCs, and proposing exact and heuristic approaches based on classical vehicle routing models;
- Oliveira (2007) and Cinar et al. (2017), considering PCs based on the order in which clients will be visited by trucks;
- Žulj et al. (2018), considering PCs based on item weights, and proposing an exact approach based on the combination of two subtours, the first one collecting only heavy items, and the second one dedicated only to light items retrieval.

2.3 Multi-fleet and two-echelon SRPs

SRPs with a fleet of different skilled vehicles have been studied in:

- Ballestín et al. (2013, 2020), for both storage and retrieval;
- Cortés et al. (2017), only for retrieval.

Nevertheless, neither transshipment nor routing restrictions have been considered in those papers. Indeed, vehicle routing restriction within warehouses is a very rarely topic in the rich literature of SRPs. Specifically, Cinar et al. (2017), inspired by Oliveira (2007), consider a SRP where SKUs are retrieved by automatic cranes and put on collectors, where they are picked up by forklifts and loaded on trucks. Despite the two-echelon structure of the system, however, the authors only focus on forklift operations, given the sequence of crane retrievals. A job-shop formulation is described and a matheuristic ap- proach is proposed.

On the other hand, multi-echelon itineraries for items to retrieve or store are often encountered when dealing with specific automatic parts-to-picker warehousing systems, such as the shuttle-based ones. Research on such systems is largely addressed, see the recent contributions of Tappia et al. (2019); Küçükyaşar et al. (2020); Wang et al. (2020); Zhao et al. (2020).

2.4 Positioning our problem with respect to the literature

The problem considered in this paper shares some features with the problems presented in this review. Nevertheless, they have never been considered jointly in a unique setting. Firstly, regarding the PCs, the above mentioned contributions only consider PCs to construct picking routes. To the best of the author's knowledge, PCs have never been discussed for storage operations which often need to be planned by respecting some precedence criteria. Moreover, in the literature, PCs are only considered for subsets of products to pick, in particular only those included in a batch (which often correspond to a single order) entrusted to a picker. On the other hand, because of the specificity of the precedence criterion addressed in our problem, PCs are applied in a broader perspective, by considering the operations of all workers and all the product types stored in the warehouse simultaneously when sequencing operations and designing routes.

Regarding fleet considerations, the literature may count on very few contributions when a not homogeneous fleet of vehicles or routing restrictions within the warehouse come into play. The latter, in particular, does not seem to have been addressed till now, except for the above mentioned unique contribution, which however focuses exclusively on the management of operations of a single part of the warehouse, while considering approximations on the operations of the other part (Cinar et al., 2017). Indeed, routing restrictions, such as for instance two-echelon routing, have been explored in contexts near SRP, particularly in the city logistics one (see for instance Crainic et al., 2009; Hemmelmayr et al., 2012). These problems, however, are different in that they do not consider demand rates, multiple vehicles per route, and multiple visits to a same location as routing workers within a warehouse normally do.

3 The problem addressed

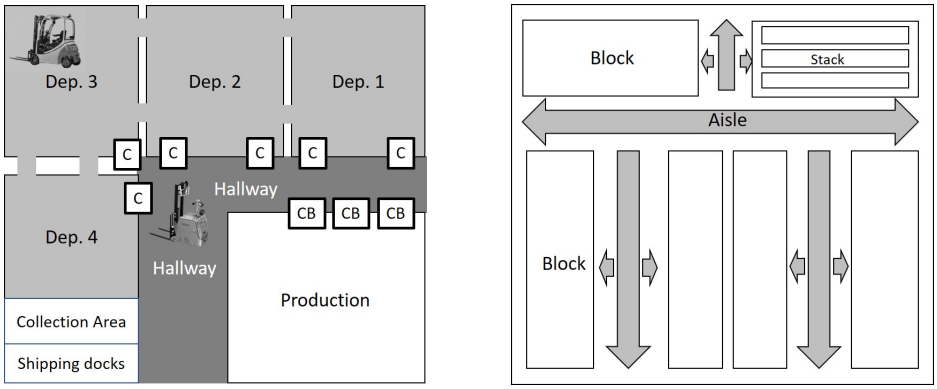
The problem is defined in a warehouse characterized by two disjoint areas. The first area is a transit zone (e.g., a large hallway) connecting the input points of the warehouse, where items wait to be stored (e.g., end-of-line conveyor belts as in the case study addressed), to the storage area; the second one is the storage area, possibly arranged in several separated departments, where items are stocked in storage locations (e.g., stacks as in the case study addressed). In

each storage location, items are homogeneously stocked with respect to the type of product. In this area, also the output point of the warehouse is located, which is a collection area where, according to the pick-and-sort policy followed, retrieved items are gathered to establish order integrity before loading trucks. Storage locations have different capacities, depending on their location within the warehouse, and both the deposit and the collection area are capacitated.

During a specified time horizon, a number of items of different product types is placed on the deposit and require the transportation to their preassigned storage locations to replenish the stock. We define this flow of items as the *incoming flow*. At the same time, a possibly different number of items needs to be picked from the storage locations and transported to the collection area to meet customer demands. We define this flow of items as the *outgoing flow*. Incoming items are available at the deposit at a known *availability date*, while outgoing items are required to reach the collection area before a known *due date*. The number of items and the product type of incoming and outgoing flows are known in advance and they are described in a *storing list* and a *shipping list*, respectively.

The movements of items are performed by capacitated vehicles belonging to two different types of fleets, defined in the following as F1 and F2. The routing of the two fleets of vehicles is restricted to only one of the above described disjoint areas of the warehouse. In particular, F1 can only move in the transit zone, whereas F2 can only circulate within the storage area. Vehicles may exchange freight at specific capacitated zones, called *collectors*. Items may hold on collectors with no time restrictions. Thus, incoming freight need to follow a two-echelon movement towards their storage locations. In fact, items are picked up from the deposit by a vehicle of type F1 and transported to one of the available collectors, where they are unloaded. From there, items are loaded by a vehicle of type F2 that moves them to their preassigned storage locations, where they are stored. The movement of outgoing freight is straightforward and consists of items loaded by a vehicle of type F2 from their storage locations and transported to the collection area. Nevertheless, they may idle on some collectors, once retrieved, and be transshipped from one vehicle to another one of type F2, before reaching destination. In addition, the routing of the vehicles has to be planned by considering: i) anticipation of outgoing movements with respect to the planned due dates, ii) a particular FIFO picking and put-away policy, and iii) safety requirements for workers, as better described next. Specifically, given the high number of operations expected during each period of the planning horizon, a crucial point for the company is to anticipate some movements related to the outgoing flow of items, to ease the movements during subsequent periods. For instance, items planned to leave the site in the second period may be moved towards the collection area during the first period. This is particularly relevant when periods with a low demand are followed by periods with high demand.

Moreover, a strict management policy has to be pursued for both picking and put-away operations, separately per product type. That is, for each product type in the storing or shipping list, the operations of filling or emptying storage locations, respectively, have to follow a prespecified *order of precedence*. More in detail, for each product type in the storage list, a set of storage locations where the items have to be stored is provided alongside with the order of precedence with which such storage locations have to be filled. Consequently, separately per product type, storage locations have to be filled up one at a time following the given order of precedence, implying that a new location may be utilized for storing only if the previous one in the considered order is already completely full. This order of precedence has to be followed also when items have to be retrieved, thus generating a strict FIFO policy to follow during picking operations, again separately per product type. A motivation to consider such a retrieval order of precedence is due to the perishability of the products stored and managed in the warehouse, like in the application context considered, with the need to retrieve and ship first the items of



(a) Warehouse representation (b) Department representation

Figure 1: Warehouse and department representation.

a given product type with the highest time of permanence within the warehouse.

Finally, given the very high number of movements within the warehouse and the need that multiple vehicles work simultaneously, vehicle congestion may inevitably occur when routes are not carefully planned. Therefore, to avoid delays in warehousing operations and to guarantee security to workers, preventing congestion becomes an issue to keep in consideration. In particular, crossing and overtaking among vehicles is allowed, but no two vehicles may travel from the same location toward another same location at the same time.

A possible structure of the warehouse is depicted in Figure 1a. The positions of the input points (denoted with CB) and of the collectors (denoted with C) are also reported. The areas of the warehouse are filled with different colors, namely dark grey and light grey, indicating the areas where vehicles of fleet F1 and F2 are allowed to move, respectively. Figure 1b shows a possible internal layout of a department, organized into blocks of stacks.

4 Mathematical models

We next describe two mathematical formulations to the problem addressed.

The first formulation, named the *storage location formulation* and presented in Section 4.1, is defined on a graph, describing the physical network on which vehicles operate in the warehouse, where each storage location is represented by a distinct node.

Since the dimension of such a model may rapidly raise as the number of the storage locations increases, in Section 4.2 we describe an alternative formulation, defined on a more compact graph, where nodes are not associated with the individual storage locations, but with groups of contiguous storage locations, with sequential priority, which are either occupied or assigned for storing to a same product type. We refer to such a group of contiguous storage locations as a *super-storage location* (SSL for short), and name such a formulation as the *super-storage location formulation*.

The complete notation (sets, parameters and variables) used to formulate both models is summarized in Table 1.

4.1 The storage location formulation

Let $\mathcal{K}$ be the set of the product types, or commodities, requiring movement in a given time horizon. The set $\mathcal{K}$ is composed of two subsets: the subset of the incoming commodities $\mathcal{K}_{in}$ and the subset of the outgoing commodities $\mathcal{K}_{out}$ (notice that $\mathcal{K}_{in}$ and $\mathcal{K}_{out}$ are not necessarily

disjoint). Let $\mathcal{V}$ be the set of vehicles in charge of moving commodities inside the warehouse. It is composed of two subsets: the subset of vehicles belonging to fleet type F1 and the subset of vehicles belonging to fleet type F2, defined as $\mathcal{V}^1$ and $\mathcal{V}^2$, respectively.

Let $\mathcal{G}^P = (\mathcal{N}^P, \mathcal{A}^P)$ be the directed graph representing the physical network on which vehicles operate in the warehouse. Specifically, the set of nodes $\mathcal{N}^P$ defines the relevant locations of the warehouse and it includes:

- the storage locations which are pertinent to the optimization process;
- the parking areas for vehicles type F1 and F2, denoted by ω^1 and ω^2, respectively;
- the set $\mathcal{R}$ of the input points which are present in the transit area, for instance conveyors;
- the set $\mathcal{B}$ of the collectors;
- the output point (or collection area) π.

In particular, not all the storage locations of the warehouse are represented in $\mathcal{N}^P$, but only those preassigned to product types in $\mathcal{K}_{in}$, and those occupied by items of product types in $\mathcal{K}_{out}$ at the beginning of the time horizon. Hereafter, $\mathcal{S}_{in}^k$ will denote the set of storage locations in which items of product type $k \in \mathcal{K}_{in}$ have to be stored, while $\mathcal{S}_{out}^k$ will denote the set of storage locations from which items of product type $k \in \mathcal{K}_{out}$ have to be retrieved. In addition, we also define $\mathcal{S}_{in}$ as the set of all storage locations assigned to all the products $k \in \mathcal{K}_{in}$ (i.e., $\mathcal{S}_{in} = \bigcup_{k \in \mathcal{K}_{in}} \mathcal{S}_{in}^k$), $\mathcal{S}_{out}$ as the set of all storage locations occupied by all the products $k \in \mathcal{K}_{out}$ (i.e., $\mathcal{S}_{out} = \bigcup_{k \in \mathcal{K}_{out}} \mathcal{S}_{out}^k$), and $\mathcal{S}^k$ as the set of all storage locations occupied by or assigned to the products $k \in \mathcal{K}$.

As previously described, precedence relationships are associated with the set of storage locations assigned to each product type $k \in \mathcal{K}_{in}$, and with the set of storage locations occupied by each product type $k \in \mathcal{K}_{out}$, defining the order of precedence according to storage and retrieval operations are allowed to be performed, respectively, per product type.

Regarding the set of the arcs $\mathcal{A}^P$, an arc (i, j) represents a direct connection between the location $i \in \mathcal{N}^P$ and the location $j \in \mathcal{N}^P$. The time to travel from i to j along (i, j), say $\tau_{i,j}$, is determined by considering the allowed speed of the vehicles and the Manhattan distance, assuming that vehicles always follow a shortest path from i to j along the network.

We model the dynamics of the problem through a *space-time network* $\mathcal{G} = (\mathcal{N}, \mathcal{A})$. Specifically, we discretize the time horizon into T time periods of equal length through $T + 1$ time instants. The set $\mathcal{N}^P$ is then replicated $T + 1$ times, resulting in set $\mathcal{N}$. A node in $\mathcal{N}$ is defined by a couple (i, t), with $i \in \mathcal{N}^P$ and $t \in \{0, \ldots, T\}$, and represents one of the locations of the warehouse at one of the considered $T + 1$ time instants. The set of arcs $\mathcal{A}$ is composed of two subsets: the subset $\mathcal{A}^H$ of the *holding* arcs and the subset $\mathcal{A}^M$ of the *moving* arcs. The subset $\mathcal{A}^H$ includes arcs of type $((i, t), (i, t+1))$, for any $i \in \mathcal{N}^P$ and $t \in \{0, \ldots, T-1\}$, which are used to model *idle time* of items or vehicles in a given node for one time period, while the subset $\mathcal{A}^M$ includes arcs of type $((i, t), (j, t'))$, with $i, j \in \mathcal{N}^P$, $i \neq j$ and $t < t'$, which are used to model *movements* of items or vehicles between two different locations in different time periods. An arc $((i, t), (j, t'))$ exists in $\mathcal{A}^M$ only if in the physical network it is possible to move from $i \in \mathcal{N}^P$ to $j \in \mathcal{N}^P$. Accordingly, $t' - t = \tau_{i,j}$. We also define four subgraphs: $\mathcal{G}_{in} = (\mathcal{N}_{in}, \mathcal{A}_{in})$, $\mathcal{G}_{out} = (\mathcal{N}_{out}, \mathcal{A}_{out})$, $\mathcal{G}_{F1} = (\mathcal{N}_{F1}, \mathcal{A}_{F1})$ and $\mathcal{G}_{F2} = (\mathcal{N}_{F2}, \mathcal{A}_{F2})$, where commodities $k \in \mathcal{K}_{in}$ and $k \in \mathcal{K}_{out}$, and vehicles $v \in \mathcal{V}^1$ and $v \in \mathcal{V}^2$ may move, respectively. The complete definition of such subgraphs is given in Appendix A.

The production is defined through parameters $d_{in}^k(r, t)$, which represent the number of items of product type $k \in \mathcal{K}_{in}$ released on $r \in \mathcal{R}$ at time t to transport toward the preassigned

locations, while the customers demand is defined by $d^k_{out}(\pi, t)$, which represent the number of items of product type $k \in \mathcal{K}_{out}$ requested in the collection area π at the latest time t.

A capacity is associated with each location of the warehouse: c_s represents the capacity of the storage location $s \in \mathcal{S}_{in} \cup \mathcal{S}_{out}$, c_r the capacity of input point $r \in \mathcal{R}$, c_π the capacity of the collection area π, and c_b the capacity of the collector $b \in \mathcal{B}$. Moreover, c_{F1} and c_{F2} represent the capacities of the vehicles type F1 and F2, respectively.

The initial state of the warehouse is defined through parameters u^k_r, u^k_b and u^k_π, for any $k \in \mathcal{K}$, $r \in \mathcal{R}$ and $b \in \mathcal{B}$, which define the number of items of product k positioned on input point r, collector b and collection area π, respectively, at the beginning of the time horizon.

In order to model the anticipation of movements of items of product types $k \in \mathcal{K}_{out}$ from the storage area towards the collection area, we need to introduce some additional parameters. The goal of such movements is to account for demands of $k \in \mathcal{K}_{out}$ beyond the considered time horizon, in order to relieve the amount of such operations in the future. We thus define an *anticipation of movements* time horizon $\tilde{T} > T$, which specifies the time periods $\tilde{t}$ beyond T, whose demand has to be preferable moved towards the collection area π before T.

Finally, we denote by $\mathcal{N}^+(i)$ and $\mathcal{N}^-(i)$ the sets of nodes linked to $i \in \mathcal{N}$ via an exiting and an entering arc, respectively, that is

$$\mathcal{N}^+(i) = \{j \in \mathcal{N} : \exists\, (i,j) \in \mathcal{A}\}, \qquad \mathcal{N}^-(i) = \{j \in \mathcal{N} : \exists\, (j,i) \in \mathcal{A}\}.$$

Now, let us define the two main families of variables which will be used:

- $x^v_{(i,t)(j,t')} \in \{0,1\}$, for any $v \in \mathcal{V}^1$ and $((i,t),(j,t')) \in \mathcal{A}_{F1}$, and for any $v \in \mathcal{V}^2$ and $((i,t),(j,t')) \in \mathcal{A}_{F2}$, which indicates whether vehicle v passes on the arc $((i,t),(j,t'))$, or not;
- $y^k_{(i,t)(j,t')} \in \mathbb{Z}_+$, for any $k \in \mathcal{K}_{in}$ and $((i,t),(j,t')) \in \mathcal{A}_{in}$, and for any $k \in \mathcal{K}_{out}$ and $((i,t),(j,t')) \in \mathcal{A}_{out}$, which indicates the number of items of product type k passing on the arc $((i,t),(j,t'))$.

In addition, we introduce two families of auxiliary variables related to picking and storing policies:

- $\alpha(s^k,t) \in \{0,1\}$, for any $s^k \in \mathcal{S}^k_{in}$ and $k \in \mathcal{K}_{in}$, and $t = 0,\dots,T$, which indicates whether the storage location s^k may be used at time t to stock product type k ($\alpha(s^k,t) = 1$), or not ($\alpha(s^k,t) = 0$);
- $\beta(s^k,t) \in \{0,1\}$, for any $s^k \in \mathcal{S}^k_{out}$ and $k \in \mathcal{K}_{out}$, and $t = 0,\dots,T$, which indicates whether the storage location s^k may be used at time t to pick up product type k ($\beta(s^k,t) = 1$), or not ($\beta(s^k,t) = 0$).

Due to its complexity, the proposed ILP model is presented for groups of constraints, starting from the objective function.

Objective function

$$\begin{aligned}\min \quad & \sum_{v\in\mathcal{V}^1} \sum_{\substack{((i,t),(j,t'))\in\mathcal{A}_{F1}:\\ i\neq\omega^1,\, j\neq\omega^1}} \tau_{i,j}\, x^v_{(i,t)(j,t')} + \sum_{v\in\mathcal{V}^2} \sum_{\substack{((i,t),(j,t'))\in\mathcal{A}_{F2}:\\ i\neq\omega^2,\, j\neq\omega^2}} \tau_{i,j}\, x^v_{(i,t)(j,t')} \\ & +\psi \sum_{k\in\mathcal{K}_{in}} \sum_{\substack{((i,t),(j,t'))\in\mathcal{A}_{in}:\\ i,\, j\in\mathcal{R}}} y^k_{(i,t)(j,t')} + \xi \sum_{k\in\mathcal{K}_{out}} P^k. \end{aligned} \tag{1}$$

The objective function is composed of four parts. The first two summations define the primary optimization goal, i.e., minimizing the travel time of all the vehicles within the warehouse. Notice that arcs entering or leaving the parking areas are not considered for both vehicles types. This is to encourage vehicles to come back to their parking areas when idle, so limiting congestion situations along the network. The third and forth summations define soft objectives. In particular, the third summation relates to the time of permanence of the items on the input points, so as to favour the movements of items towards other spots of the warehouse. The fourth relates to the anticipation movements to perform. The latter summations are weighted through parameters ψ and ξ, respectively, both to state their mutual priorities as well as their priority with respect to the primary optimization goal, and also to allow a comparison among the different units of measure of soft and primary criteria, through a suitable parameter calibration. In particular, the terms P^k are defined as follows:

$$P^k = \max\left\{0,\ \sum_{t=0}^{\tilde{T}} d^k_{out}(\pi,t) - \left[u^k_\pi + \sum_{t=0}^{T}\sum_{(j,t')\in N^-(\pi,t)} y^k_{(j,t')(\pi,t)}\right]\right\} \tag{2}$$

for any $k \in \mathcal{K}_{out}$. The rationale of this penalty is to compare the amount of items of type $k \in \mathcal{K}_{out}$ in the collection area, given by the last two addendum of (2) (i.e., the amount of items at the beginning of the time horizon, u^k_π, plus the items transported to π during the time horizon), with the overall demand of k, i.e., from the time instant $t = 0$ to the extended time instant $\tilde{T}$, given by the first addendum of (2). The penalty term is equal to 0 if during the considered time horizon an amount of items of product k enough to satisfy the overall demand of k (i.e., in the time horizon and in the extended one) is moved to the collection area. Otherwise, the penalty to be paid is set proportionally to the amount of future demand that cannot be moved in advance.

Vehicle routing constraints

$$\sum_{(j,t')\in\mathcal{N}^+(i,t)} x^v_{(i,t)(j,t')} - \sum_{(j,t')\in\mathcal{N}^-(i,t)} x^v_{(j,t')(i,t)} = \begin{cases} 1 & \text{if } (i,t) = (\omega^1, 0), \\ -1 & \text{if } (i,t) = (\omega^1, T), \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$
$$\forall\ (i,t) \in \mathcal{N}_{F1},\ \forall\ v \in \mathcal{V}^1,$$

$$\sum_{(j,t')\in\mathcal{N}^+(i,t)} x^v_{(i,t)(j,t')} - \sum_{(j,t')\in\mathcal{N}^-(i,t)} x^v_{(j,t')(i,t)} = \begin{cases} 1 & \text{if } (i,t) = (\omega^2, 0), \\ -1 & \text{if } (i,t) = (\omega^2, T), \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$
$$\forall\ (i,t) \in \mathcal{N}_{F2}, \forall\ v \in \mathcal{V}^2,$$

$$\sum_{v\in\mathcal{V}^1} x^v_{(i,t)(j,t')} \le 1 \qquad \forall\ ((i,t),(j,t')) \in \mathcal{A}_{F1} :\ i,j \neq \omega^1, \tag{5}$$

$$\sum_{v\in\mathcal{V}^2} x^v_{(i,t)(j,t')} \le 1 \qquad \forall\ ((i,t),(j,t')) \in \mathcal{A}_{F2} :\ i,j \neq \omega^2. \tag{6}$$

Constraints (3) and (4) ensure the correctness of the routing of the vehicles. Recalling that vehicles may only move in their respective subgraphs, (3) and (4) state that vehicles of F1 and F2 have to start their route from their parking areas (i.e., ω^1 or ω^2, respectively) at the beginning of the time horizon (i.e., at $t = 0$), and have to return there at the end of the

time horizon (i.e., at $t = T$). The before mentioned security requirements are worded through constraints (5) and (6), by imposing that at most one vehicle, either of F1 or of F2, can be present in any arc of their respective subgraph. The only exceptions are for the holding arcs representing dwell time at their respective parking areas.

Incoming freight flow constraints

$$\begin{aligned}
&\sum_{(j,t')\in\mathcal{N}^+(i,t)} y^k_{(i,t)(j,t')} - \sum_{(j,t')\in\mathcal{N}^-(i,t)} y^k_{(j,t')(i,t)} \\
&\qquad = \begin{cases} d^k_{in}(i,t) + u^k_i & \text{if } i \in \mathcal{R},\ t = 0, \\ u^k_i & \text{if } i \in \mathcal{B},\ t = 0, \\ d^k_{in}(i,t) & \text{if } i \in \mathcal{R},\ t = 1,\dots,T, \\ 0 & \text{if } i \in \mathcal{B}, t = 1,\dots,T, \\ 0 & \text{if } i \in \mathcal{S}^k_{out} \cup \mathcal{S}^{k'},\ t = 0,\dots,T, \end{cases} \\
&\forall\, k \in \mathcal{K}_{in},\ \forall\, k' \in \mathcal{K} :\ k' \neq k, \\
&\forall\, (i,t) \in \mathcal{N}_{in} :\ i \in \mathcal{R} \cup \mathcal{B} \cup \mathcal{S}^k_{out} \cup \mathcal{S}^{k'}.
\end{aligned} \tag{7}$$

Constraints (7) are the flow conservation constraints for the incoming product types $k \in \mathcal{K}_{in}$. New releases during the time horizon are represented by values $d^k_{in}(r,t) > 0$ for some r and time instant t. For $t = 0$, it is considered the chance of already having some items idling on some r or some b, as a results of operations previously performed. Also notice that items of a product type $k \in \mathcal{K}_{in}$ can never be put in a storage location other than the one preassigned to k. The flow of each product $k \in \mathcal{K}_{in}$ thus always terminates in one of its preassigned storage locations.

Outgoing freight flow constraints

$$\begin{aligned}
&\sum_{(j,t')\in\mathcal{N}^+(i,t)} y^k_{(i,t)(j,t')} - \sum_{(j,t')\in\mathcal{N}^-(i,t)} y^k_{(j,t')(i,t)} = \begin{cases} u^k_i & \text{if } i \in \mathcal{B}, t = 0, \\ 0 & \text{if } i \in \mathcal{B}, t \geq 1, \\ 0 & \text{if } i \in \mathcal{S}^k_{out} \cup \mathcal{S}^{k'} \\ & \text{and } t \geq 0, \end{cases} \\
&\qquad\qquad \forall\ k \in \mathcal{K}_{out},\ \forall\ k' \in \mathcal{K} :\ k \neq k', \\
&\qquad\qquad \forall\ (i,t) \in \mathcal{N}_{out} :\ i \in \mathcal{B} \cup \mathcal{S}^{k'} \cup \mathcal{S}^k_{in},
\end{aligned} \tag{8}$$

$$\begin{aligned}
&\sum_{(j,t')\in\mathcal{N}^-(\pi,t)} y^k_{(j,t')(\pi,t)} - \sum_{(\pi,t')\in\mathcal{N}^+(\pi,t)} y^k_{(\pi,t)(\pi,t')} = d^k_{out}(\pi,t) \\
&\qquad\qquad \forall\ k \in \mathcal{K}_{out},\ \forall\ t \geq 1,
\end{aligned} \tag{9}$$

$$y^k_{(\pi,0)(\pi,1)} = u^k_\pi \qquad\qquad \forall\ k \in \mathcal{K}_{out}. \tag{10}$$

Relations (8) are the flow conservation constraints for $k \in \mathcal{K}_{out}$. As for the case of the incoming freight flow, it is considered the chance of having some items of product type $k \in \mathcal{K}_{out}$ idling on some b at time $t = 0$, as a result of operations previously performed. Moreover, items of a product type $k \in \mathcal{K}_{out}$ can never be stored in any storage location once retrieved. Relations (9)–(10) are demand constraints. In particular, constraints (9) ensure that all the items of product type $k \in \mathcal{K}_{out}$ requested at time t are transported to the collection area before t, while (10) defines the composition of the collection area at the beginning of the time horizon.

Linking capacity constraints

$$\sum_{\substack{k\in\mathcal{K}_{in}:\\((i,t),(j,t'))\in\mathcal{A}_{in}}} y^k_{(i,t)(j,t')} \le c_{F1} \sum_{v\in\mathcal{V}^1} x^v_{(i,t)(j,t')} \qquad \forall\ ((i,t),(j,t'))\in\mathcal{A}_{F1}, \tag{11}$$

$$\sum_{\substack{k\in\mathcal{K}_{in}:\\((i,t),(j,t'))\in\mathcal{A}_{in}}} y^k_{(i,t)(j,t')} + \sum_{\substack{k\in\mathcal{K}_{out}:\\((i,t),(j,t'))\in\mathcal{A}_{out}}} y^k_{(i,t)(j,t')} \le c_{F2} \sum_{v\in\mathcal{V}^2} x^v_{(i,t)(j,t')} \qquad \forall\ ((i,t),(j,t'))\in\mathcal{A}_{F2}. \tag{12}$$

Relations (11)–(12) define the linking capacity constraints for vehicles of F1 and F2, respectively, by considering both incoming and outgoing items flows. In particular, they state that freight flows can only be transported by means of vehicles which have been selected to move within the warehouse, and that the total commodity flow on any moving arc cannot exceed the capacity of the vehicle traveling along it.

Location capacity constraints

$$\sum_{k\in\mathcal{K}_{in}} d^k_{in}(r,t) + \sum_{k\in\mathcal{K}_{in}} y^k_{(r,t-1)(r,t)} \le \begin{cases} c_r - \sum\limits_{k\in\mathcal{K}_{in}} u^k_r & \text{if } t=1,\\ c_r & \text{if } t>1, \end{cases} \qquad \forall\ r\in\mathcal{R},\ \forall\ t\ge 1, \tag{13}$$

$$\sum_{(j,t')\in\mathcal{N}^-(b,t)} \sum_{k\in\mathcal{K}} y^k_{(j,t')(b,t)} \le \begin{cases} c_b - \sum\limits_{k\in\mathcal{K}} u^k_b & \text{if } t=1,\\ c_b & \text{if } t>1, \end{cases} \qquad \forall\ b\in\mathcal{B},\ \forall\ t\ge 1, \tag{14}$$

$$\sum_{(j,t')\in\mathcal{N}^-(\pi,t)} \sum_{k\in\mathcal{K}_{out}} y^k_{(j,t')(\pi,t)} \le \begin{cases} c_\pi - \sum\limits_{k\in\mathcal{K}_{out}} u^k_\pi & \text{if } t=1,\\ c_\pi & \text{if } t>1, \end{cases} \qquad \forall\ t\ge 1, \tag{15}$$

$$\sum_{\tilde{t}=0}^{t} \left[\sum_{(j,t')\in\mathcal{N}^-(i,\tilde{t})} y^k_{(j,t')(i,\tilde{t})} - \sum_{(j,t')\in\mathcal{N}^+(i,\tilde{t})} y^k_{(i,\tilde{t})(j,t')} \right] \le c_i \qquad \forall\ k\in\mathcal{K}_{in},\ \forall\ t\ge 1,\ \forall\ (i,t)\in\mathcal{N}_{in}:\ i\in\mathcal{S}^k_{in}, \tag{16}$$

$$\sum_{\tilde{t}=0}^{t} \left[\sum_{(j,t')\in\mathcal{N}^+(i,\tilde{t})} y^k_{(i,\tilde{t})(j,t')} - \sum_{(j,t')\in\mathcal{N}^-(i,\tilde{t})} y^k_{(j,t')(i,\tilde{t})} \right] \le c_i \qquad \forall\ k\in\mathcal{K}_{out},\ \forall\ t\ge 1,\ \forall\ (i,t)\in\mathcal{N}_{out}:\ i\in\mathcal{S}^k_{out}. \tag{17}$$

Relations (13)–(17) define the capacity constraints for each location of the warehouse. In particular, constraints (13) relate to input points, constraints (14) relate to collectors, and constraints (15) relate to the collection area. Moreover, by considering the incoming flow, constraints (16)

guarantee the satisfaction of the capacity of each storage location preassigned to $k \in \mathcal{K}_{in}$. Similarly, by considering the outgoing flow, constraints (17) state the maximum number of items that can be retrieved from storage locations occupied by product types $k \in \mathcal{K}_{out}$.

Storage policy constraints

$$\sigma^t_{s^k} = \sum_{\tilde{t}=0}^{t} \left[\sum_{(j,t') \in \mathcal{N}^-(s^k,\tilde{t})} y^k_{(j,t')(s^k,\tilde{t})} - \sum_{(j,t') \in \mathcal{N}^+(s^k,\tilde{t})} y^k_{(s^k,\tilde{t})(j,t')} \right] \quad \forall\, k \in \mathcal{K}_{in},\ \forall\, s^k \in \mathcal{S}^k_{in},\ \forall\, t \geq 1, \tag{18}$$

$$c_{s^k_l} - \sigma^t_{s^k_l} \leq c_{s^k_l} \left[1 - \alpha(s^k_{l+1}, t)\right] \quad \forall\, k \in \mathcal{K}_{in},\ \forall\, t = 0, \ldots, T,\ \forall\, s^k_l \in \mathcal{S}^k_{in},\ \forall\, l = 1, \ldots, |\mathcal{S}^k_{in}| - 1, \tag{19}$$

$$\sum_{(j,t') \in \mathcal{N}^-(s^k,t)} y^k_{(j,t')(s^k,t)} \leq c_{s^k}\, \alpha(s^k, t) \quad \forall\, k \in \mathcal{K}_{in},\ \forall\, s^k \in \mathcal{S}^k_{in},\ \forall\, t \geq 0. \tag{20}$$

In accordance with the specific storage policy, storage locations are required to be filled sequentially by respecting the specified order of precedence. For example, let $s^k_1 \in \mathcal{S}^k_{in}$ be the first storage location eligible for stocking the product type $k \in \mathcal{K}_{in}$ and $s^k_2 \in \mathcal{S}^k_{in}$ be the second storage location eligible for stocking (in accordance to the order of precedence of the preassigned storage locations). At the beginning of the time horizon, i.e., at $t = 0$, stocking has to begin from s^k_1 and then continue, only once it is full, by using the next preassigned storage location, i.e., s^k_2. Constraints (18)–(20) state this policy. In particular, equations (18) define the total number of items of product type $k \in \mathcal{K}_{in}$ stocked in the storage location $s^k \in \mathcal{S}^k_{in}$ until time t (note that, at $t = 0$ and for the first storage location in the given order of precedence, this is an input data). If storage location s^k_l has not already reached its saturation at time t, constraints (19) do not allow the next assigned storage location in the related order of precedence, i.e., s^k_{l+1}, to be used to stock items of product type $k \in \mathcal{K}_{in}$: this is mathematically guaranteed by forcing $\alpha(s^k_{l+1}, t) = 0$ in this scenario thanks to constraints (19). As opposed, when storage location s^k_l has reached its saturation, i.e., $c_{s^k_l} = \sigma^t_{s^k_l}$, storage location s^k_{l+1} becomes eligible to stock items of product type k, being $\alpha(s^k_{l+1}, t)$ allowed by the combination of constraints (19) and (20) to assume value 1, that is $\alpha(s^k_{l+1}, t) = 1$.

Retrieval policy constraints

$$\rho^t_{s^k} = \sum_{\tilde{t}=0}^{t} \left[\sum_{(j,t') \in \mathcal{N}^+(s^k,\tilde{t})} y^k_{(s^k,\tilde{t})(j,t')} - \sum_{(j,t') \in \mathcal{N}^-(s^k,\tilde{t})} y^k_{(j,t')(s^k,\tilde{t})} \right] \quad \forall\, k \in \mathcal{K}_{out},\ \forall\, s^k \in \mathcal{S}^k_{out},\ \forall\, t \geq 1, \tag{21}$$

$$c_{s^k_l} - \rho^t_{s^k_l} \leq c_{s^k_l} \left(1 - \beta(s^k_{l+1}, t)\right) \quad \forall\, k \in \mathcal{K}_{out},\ \forall\, t \geq 0,\ \forall\, s^k_l \in \mathcal{S}^k_{out},\ \forall\, l = 1, \ldots, |\mathcal{S}^k_{out}| - 1, \tag{22}$$

$$\sum_{(j,t') \in \mathcal{N}^+(s^k,t)} y^k_{(s^k,t)(j,t')} \leq c_s\, \beta(s^k, t) \quad \forall\, k \in \mathcal{K}_{out},\ \forall\, s^k \in \mathcal{S}^k_{out},\ \forall\, t \geq 0. \tag{23}$$

In accordance with the specific retrieval policy, storage locations are required to be emptied sequentially by respecting their specified order of precedence. Constraints (21)–(23), whose logic is similar to constraints (18)–(20), state this policy. In particular, equations (21) define the total number of items of product type $k \in \mathcal{K}_{out}$ retrieved from the storage location $s^k \in \mathcal{S}^k_{out}$ until time t (also in this case, at $t = 0$ and for the first storage location in the given order of precedence, this is an input data). Constraints (22) impose that the next storage location in the related order of precedence, s^k_{l+1}, cannot be used to retrieve items of product type k, unless the previous storage location, s^k_l, has been completely emptied. In the latter case, $\beta(s^k_{l+1}, t) = 1$ is allowed by the combination of constraints (22) and (23); otherwise $\beta(s^k_{l+1}, t) = 0$ and retrieval still has to be performed from s^k_l.

4.2 The super-storage location formulation

As mentioned before, the dimension of the model presented in Section 4.1 may rapidly raise as the number of the storage locations pertinent to the optimization process increases. Thus, we also present a model based on SSLs.

The capacity of any SSL is the sum of the capacities of the single storage locations composing it. Denoting with $\tilde{\mathcal{S}}^k_{in}$ the set of the SSLs assigned to a product type $k \in \mathcal{K}_{in}$ for storing operations, with $\tilde{\mathcal{S}}^k_{out}$ the set of the SSLs occupied by a product type $k \in \mathcal{K}_{out}$, and with $\tilde{\mathcal{S}}_{in}$ and $\tilde{\mathcal{S}}_{out}$, respectively, the set of the SSLs assigned to all the products $k \in \mathcal{K}_{in}$ and the set of the SSLs occupied by all the products $k \in \mathcal{K}_{out}$, then we define the capacity of $\tilde{s} \in \tilde{\mathcal{S}}_{in} \cup \tilde{\mathcal{S}}_{out}$ as $\tilde{c}_{\tilde{s}}$. Since each SSL is defined by grouping SLs with sequential priority for storage or retrieval operations, that is, SLs which have to be emptied or replenished one after the other according to the given precedence relationships among the SLs, then the precedence relationships among the SSLs can be derived straightforwardly starting from the precedence relationships among the SLs.

Using the notation introduced above, the resulting super-storage location formulation can be obtained from model (1)–(23) by appropriately replacing the sets and the parameters associated with storage locations with those associated with super-storage locations.

Notice that, if on the one hand the alternative representation of the warehouse in terms of SSLs may bring to a reduction of the dimension of the associated graph, and thus ease the resolution process, on the other hand this may lead to a less manageable solution, since workers have now information about storing or picking operations not at a storage location level, but rather at a super-storage location level.

5 Matheuristic approach

For real instances, such as those provided to us by our industrial partner, the proposed formulations may have a very high dimension because of the huge number of products and storage locations involved in storing and retrieving operations (recall that we address warehouses with a high degree of product rotation). Thus, the models cannot be directly addressed through state-of-the-art commercial solvers like CPLEX. Therefore, we propose a matheuristic approach based on a decomposition strategy. Specifically, the planning horizon is divided into Λ subperiods, by splitting the original time horizon into Λ periods of equal (or different) length. Each subperiod thus gives rise to a subproblem, whose features are those of the original problem restricted to the considered subperiod. The Λ subproblems are then sequentially solved by using CPLEX, in such a way that the final state of the system obtained solving subproblem $\lambda - 1$ becomes the initial state of the system when solving subproblem λ, for any $\lambda = 2, \ldots, \Lambda$. In particular, the state of the system in each subproblem takes into account the position of vehicles and items

within the warehouse. Once the Λ subproblems have been solved, in order to construct a solution for the original problem, and thus the complete schedule for the entire time horizon, it is sufficient to concatenate the Λ solutions in an increasing order with respect to the subperiod addressed, i.e., from subperiod 1 to subperiod Λ.

The subperiod reformulations may be derived straightforwardly from the complete planning horizon formulations described in Section 4, by keeping unchanged the structure of the majority of the constraints. Nevertheless, parameter T now defines the final time instant of the generic subperiod, instead of the end of the whole time horizon. We discuss here the main modifications involving the matheuristic based on the storage location formulation. The ones for the matheuristic based on the super-storage location formulation are analogous. Specifically, constraints (3), (4), (7) and (8) in Section 4.1 need to be modified. Since in the subperiod $\lambda = 1, \ldots, \Lambda - 1$, the vehicles of type F1 are not obliged to go back to their parking area at the end of it, then denoting by $u^v \in \mathcal{N}_{F1}$ the physical node from where the vehicle $v \in \mathcal{V}^1$ begins its route in subperiod $\lambda > 1$, the set of constraints (3) is modified in the following way:

- if $\lambda = 1$, then the vehicles depart from their parking area:

$$\sum_{(j,t') \in \mathcal{N}^+(i,t)} x^v_{(i,t)(j,t')} - \sum_{(j,t') \in \mathcal{N}^-(i,t)} x^v_{(j,t')(i,t)} = \begin{cases} 1 & \text{if } (i,t) = (\omega^1, 0), \\ 0 & \text{otherwise,} \end{cases} \qquad \forall\, v \in \mathcal{V}^1,\ \forall\, (i,t) \in \mathcal{N}_{F1}; \tag{24}$$

- if $\lambda = \Lambda$, then the vehicle v departs from u^v and then returns to the parking area at the end of the subperiod:

$$\sum_{(j,t') \in \mathcal{N}^+(i,t)} x^v_{(i,t)(j,t')} - \sum_{(j,t') \in \mathcal{N}^-(i,t)} x^v_{(j,t')(i,t)} = \begin{cases} 1 & \text{if } (i,t) = (u^v, 0), \\ -1 & \text{if } (i,t) = (\omega^1, T), \\ 0 & \text{otherwise,} \end{cases} \qquad \forall\, v \in \mathcal{V}^1,\ \forall\, (i,t) \in \mathcal{N}_{F1}; \tag{25}$$

- if $1 < \lambda < \Lambda$, then the vehicle v starts its route from the node where it ended in the previous subperiod, i.e., u^v:

$$\sum_{(j,t') \in \mathcal{N}^+(i,t)} x^v_{(i,t)(j,t')} - \sum_{(j,t') \in \mathcal{N}^-(i,t)} x^v_{(j,t')(i,t)} = \begin{cases} 1 & \text{if } (i,t) = (u^v, 0), \\ 0 & \text{otherwise,} \end{cases} \qquad \forall\, v \in \mathcal{V}^1,\ \forall\, (i,t) \in \mathcal{N}_{F1}. \tag{26}$$

The same applies to the vehicles of type F2, therefore constraints (4) are similarly modified.

In addition, it needs to be considered that at the beginning of a subperiod λ some items of type $k \in \mathcal{K}$ may be in front of a storage location to which they are not assigned (just passing) as a result of operations in subperiod $\lambda - 1$. Let u^k_s be the number of items of product type $k \in \mathcal{K}$ located in front of a storage location s in $\mathcal{S}_{in} \cup \mathcal{S}_{out}$ at the beginning of subperiod λ. Constraints (7) and (8) are then modified as follows:

Algorithm 1 The matheuristic approach

1: Divide the time horizon into Λ subperiods
2: **for** $\lambda = 1, \ldots, \Lambda$ **do**
3: Solve subproblem λ
4: **end for**
5: Concatenate the subproblem solutions from 1 to Λ

- for product types in $\mathcal{K}_{in}$:

$$\sum_{(j,t')\in\mathcal{N}^+(i,t)} y^k_{(i,t)(j,t')} - \sum_{(j,t')\in\mathcal{N}^-(i,t)} y^k_{(j,t')(i,t)} = \begin{cases} d^k_{in}(i,t) + u^k_i & \text{if } i \in \mathcal{R},\ t = 0, \\ d^k_{in}(i,t) & \text{if } i \in \mathcal{R},\ t = 1, \ldots, T, \\ u^k_i & \text{if } i \in \mathcal{B},\ t = 0, \\ u^k_i & \text{if } i \in \mathcal{S}^k_{out} \cup \mathcal{S}^{k'},\ t = 0, \\ 0 & \text{if } i \in \mathcal{B} \cup \mathcal{S}^k_{out} \cup \mathcal{S}^{k'},\ t = 1, \ldots, T, \end{cases} \tag{27}$$
$$\forall\ k \in \mathcal{K}_{in},\ \forall\ k' \in \mathcal{K} :\ k' \neq k,$$
$$\forall\ (i,t) \in \mathcal{N}_{in} :\ i \in \mathcal{R} \cup \mathcal{B} \cup \mathcal{S}^k_{out} \cup \mathcal{S}^{k'};$$

- for product types in $\mathcal{K}_{out}$:

$$\sum_{(j,t')\in\mathcal{N}^+(i,t)} y^k_{(i,t)(j,t')} - \sum_{(j,t')\in\mathcal{N}^-(i,t)} y^k_{(j,t')(i,t)} = \begin{cases} u^k_i & \text{if } i \in \mathcal{B},\ t = 0, \\ u^k_i & \text{if } i \in \mathcal{S}^k_{out} \cup \mathcal{S}^{k'},\ t = 0, \\ 0 & \text{if } i \in \mathcal{B} \cup \mathcal{S}^k_{out} \cup \mathcal{S}^{k'},\ t = 1, \ldots, T, \end{cases} \tag{28}$$
$$\forall\, k \in \mathcal{K}_{out},\ \forall\, k' \in \mathcal{K} :\ k' \neq k,$$
$$\forall\, (i,t) \in \mathcal{N}_{out} : i \in \mathcal{B} \cup \mathcal{S}^{k'} \cup \mathcal{S}^k_{in}.$$

The matheuristic approach is summarized in Algorithm 1.

6 Numerical experiments

6.1 The case study addressed

The production site of the company we consider, leader in the tissue sector, is composed of a production area, a large warehouse, a collection area, and several shipping docks. The warehouse is larger than 10,000 m^2 and it is located beside the production area and connected to it by a large hallway. The warehouse is composed of four departments. Each department has a rectangular internal layout with a certain number of parallel narrow storage aisles and parallel wide cross aisles. The storage area is thus divided into blocks of storage locations framed by aisles. Items are homogeneously (with respect to the product type) stored back-to-back to each other in each storage location, in such a way to define horizontal stacks of items of the same type, accessible only frontally. A random storage policy (respecting though the homogeneity criterion) is applied. Different blocks may be composed of different number of stacks, all having

though the same capacity. However, stacks belonging to different blocks may have different capacities. Specifically, the storage area is divided into 29 blocks, which are composed of a variable number of stacks ranging from 15 to 65. Stacks have a capacity ranging from 8 to 17 items, independently on the product type to store. According to the pick-and-sort policy followed, the collection area is used to gather retrieved items and establish order integrity before loading the trucks, and it is positioned at the end of the forth department. It can stock up to 700 items, and is normally filled up as much as possible during the night to quickly start the truck loading operations the next morning.

The production site works daily on three shifts of eight hours. Production never stops during the day, while orders are shipped during the first and the second shift only. More than 300 different types of products are produced in this site. Items are released by the production on three end-of-line conveyor belts (just conveyors in the following), arranged in unit-loads and wrapped in so-called *columns* of pallets. Therefore, the inventory will be expressed in terms of columns in our study. Conveyors can hold a limited quantity of columns (precisely, 10, 14 and 8 columns, respectively) and need to be emptied as soon as possible when columns are released not to block subsequent releases (production decisions are independent, and they are not addressed here). Columns are released at a constant rate during the shift. Each release is characterized by a *release time instant*, an *amount of columns* released per product type, and the conveyor of release. Additionally, the storing list also reports, separately per product type, the set of stacks assigned to the product type for storing operations, and the order of precedence according to which they have to be filled up. The decisions on the assignment and sequencing of stacks per product type are not part of this study, and they are discussed in Lanza et al. (2022). The shipping list of a day is normally known a day in advance and reports the composition of each order, that is amount of columns and types of product requested, and the leaving time of the associated truck. Items are required to be retrieved from stacks following the given order of precedence per product type, and they are moved to the collection area before a given *due date*, not to generate truck loading delay.

The fleet of the company is composed of five LGV shuttles and seven forklifts (LGV and FKL in the following). Referring to the more general problem description in Section 3, LGV and FKL correspond to vehicles of type F1 and F2, respectively. Both types of vehicles may transport two columns at most at the same time. LGV may only move on the hallway connecting conveyors and departments, while FKL may move within the departments and the collection area. Collectors are positioned at the entrance of each department. The warehouse contains six collectors, with different capacities ranging from two to eight columns. Items may hold on collectors with no time restrictions, but generally it is preferable to move them as soon as possible towards their destination, be this a storage location or the collection area, in order to avoid congestion of items around the warehouse. Incoming items are thus moved from conveyors to collectors by LGV, possibly idling on collectors, and then moved from collectors to stacks by FKL. Outgoing items, instead, are moved from stacks to the collection area by FKL, by possibly idling on collectors as well. LGV and FKL are allowed to cross and overtake each other in their respective routing areas, but no two vehicles may travel from the same location toward another same location at the same time, to limit the congestion.

Moreover, given the high number of operations required during each shift, a crucial point for the company is to anticipate as much as possible the movements of requested items towards the collection area during a shift, to ease the work load during the subsequent shift. So, for instance, items planned to leave the site during the second shift of a day, may be moved towards the collection area already during the first shift. This is particularly needed for the third shift, where the collection area is filled up as much as possible to quickly load trucks the next morning.

As in the general presentation in Section 3, critical issues are thus to perform storage and

retrieval operations by following a strict order of precedence, to avoid vehicle congestion, and to anticipate movements for outgoing items during each shift. The layout of the warehouse and the structure of each department are depicted in Figure 1a and Figure 1b of Section 3, respectively.

6.2 Plan of the experiments

Three types of experiments have been performed, as reported below.

1. Since the storage location based formulation described in Section 4.1 (SL in the following) and the super-storage location based formulation presented in Section 4.2 (SSL in the following) could not be solved directly via the state-of-the-art optimization solver CPLEX on real size instances, due to their dimension, in the first type of experiments we performed several tests on a set of small-medium size artificial instances (see Section 6.4). The goal is threefold:

   (i) to compare the solutions obtained by solving SL and SSL with CPLEX under some alternative parameter settings, in order both to check the parameter impact on the solution quality and the algorithm performance, and to analyse what is lost by considering the more aggregated super-storage location representation instead of the original storage location configuration (Section 6.4.1);

   (ii) to evaluate the quality of the solutions obtained by the matheuristic presented in Section 5, by comparing them to the optimal solutions obtained by solving SL and SSL with CPLEX (Section 6.4.2);

   (iii) to test some relaxations of problem SRP, by removing critical sets of constraints from SL and SSL, as well as alternative versions of the artificial instances, to provide some managerial insights on what makes the addressed SRP computationally hard to solve (Section 6.4.3).

2. In the second type of experiments, we tested the efficacy and the efficiency of the matheuristic, when using either formulation SL or SSL, on a wide pool of real instances related to the addressed case study. The results are reported in Section 6.5.

3. Finally, in Section 6.6 we further investigated the efficiency and the efficacy of the matheuristic by considering as input data one of the busiest weeks for our industrial partner, where the movements of items are far beyond the annual average. This third type of analysis involves the consecutive resolution of the addressed SRP for each day of the selected week, by considering the formulation and the setting of the parameters suggested by the second type of experiments.

The artificial and the real instances are described in Section 6.3. Both the formulations as well as the matheuristic approach have been implemented using the OPL language and solved via CPLEX 12.6 solver (IBM ILOG, 2016). All the experiments have been conducted on an Intel Xeon 5120 computer with 2.20 GHz and 32 GB of RAM.

6.3 The instances

The data set provided by the company comprises the following information for a pool of selected shifts, each having a duration of eight hours:

(i) the warehouse configuration at the beginning of the shift, i.e., the product types and the corresponding number of columns inside the warehouse;

(ii) the storing list of the shift;

(iii) the shipping list of the shift and of the next three shifts.

Some data needed to be integrated, others instead, not provided by the company, were randomly generated. Specifically, the positions of the columns in the warehouse at the beginning of a shift are randomly generated by respecting some agreed industrial practice or insights given by the company, in such a way as to start with a realistic configuration. Additionally, the retrieval order of precedence per product type for the occupied stacks in the warehouse was randomly generated as well. On the other hand, the stacks assigned to each product type in the storing list and the corresponding filling order of precedence have been obtained by applying the resolution method proposed in Lanza et al. (2022). Finally, the truck leaving times have been randomly generated by considering that the majority of the orders are shipped during the morning.

In order to perform the first type of experiments, five artificial instances have been generated starting from the above described data set, but shortening the duration of a shift from eight to four hours and reducing the number of product types and columns to move. In particular, shipments with a number of columns lower than two are disregarded. For the second type of analysis, instead, 15 real shifts have been selected, by thus generating 15 corresponding instances. Finally, the third type of analysis has been performed on one of the busiest weeks for our industrial partner, as previously outlined. The features of the five artificial instances and of the 15 real instances are reported in Table 2. Specifically, the number of stacks and the corresponding number of super-stacks are reported together with the number of the product types in K_{in} and in K_{out} (columns K_{in} and K_{out}, respectively), and the corresponding number of items to move (columns C_{in} and C_{out}, respectively).

6.4 Tests on artificial instances

We present here the experiments conducted on the artificial instances.

6.4.1 Parameter setting and SL vs SSL comparison

Formulations SL and SSL rely on parameters ψ and ξ, which are related to the soft optimization criteria. Specifically, increasing the value of ψ would tend to give priority to emptying conveyors, moving columns as soon as they are released from the production area towards the collectors and thus towards the assigned storage locations. Increasing the value of ξ, instead, would tend to give priority to the anticipation movements toward the collection area. To be effective with respect to the primary optimization goal, which consists in minimizing the vehicle travel time, such parameters have to be set to a value higher than the minimum time required by the vehicles to either perform a storing or a retrieval operation, so as to favour the soft objectives. Specifically, the minimum time required to perform a storing operation is the minimum time needed by a LGV to move from its parking area to the conveyor where columns idle, load them and move them to a bay, then moving back to the parking area, plus the minimum time needed by a FKL to move from its parking area to the bay, load the columns and move them to the assigned storage locations, then returning to the parking area. On the other hand, the minimum time required by a FKL to perform a retrieval operation is the minimum time to move from its parking area to the storage locations, load columns and move them to the collection area, then moving back to the parking area. Based on these observations, the minimum value for both parameters ψ and ξ has been set equal to 10 in our experimentation. Notice that this way of determining values for ψ and ξ also enables a comparison between soft and primary goals, the latter one being the time spent by the vehicles.

We performed some preliminary tests by solving a subset of the artificial instances with several values for both ψ and ξ. Indeed, some real instances were also tested in this phase. Specifically, we set ψ and ξ to the minimum value 10, and then we increased them 10 by 10 till the value 50. For increasing values of ψ and ξ, we observed only slight improvements in terms of emptying conveyors and anticipation movements, at the expense however of a greater difficulty of CPLEX in solving the instances. In particular, setting ψ and ξ to 50 complicates a lot the resolution process. We therefore decided to solve the instances using the extreme values 10 and 50 for both ψ and ξ, in their four combinations. In the following, we refer to a weight combination as a pair of numbers in brackets of type $(\cdot$ - $\cdot)$, where the first position is associated with ψ and the second one with ξ. The tested weight combinations are thus (10-10), (10-50), (50-10) and (50-50) for both formulations SL and SSL. A time limit of one hour has been set for CPLEX.

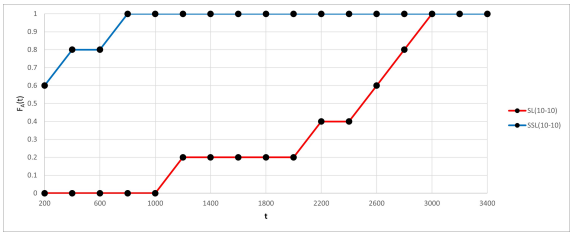
Table 3 reports the average optimality gaps and the solving times of CPLEX, expressed in seconds, and some aggregated features of the solutions obtained for the above mentioned weight combinations, separately for SL and SSL. The primary goal is analysed in terms of the average time (in minutes) travelled by a LGV and by a FKL over the 5 instances. The secondary goals, i.e., the emptying of conveyors and the anticipation moves, are evaluated considering the average time (in minutes) an incoming item idles on conveyors before been moved to an available collector over the 5 instances, and the percentage of saturation of the collection area both 60 minutes before the end of the planning horizon (% of saturation of collection area after 3h) and also at the end of the planning horizon (% of saturation of collection area after 4h).

On the artificial instances, the impact of the weight combinations on the main features of the computed solutions is not so evident, and will be better investigated for the set of the real instances. In particular, the average travel time of LGV and FKL is almost the same for all the considered weight combinations. Regarding the other features, when ξ is fixed and ψ increases, the average idle time on conveyors tends to decrease, as expected, although the variations appear to be quite marginal. When ψ is fixed and ξ increases, instead, the percentage of saturation of the collection area after three hours slightly increases, sometimes, showing a prioritization of the anticipation moves.

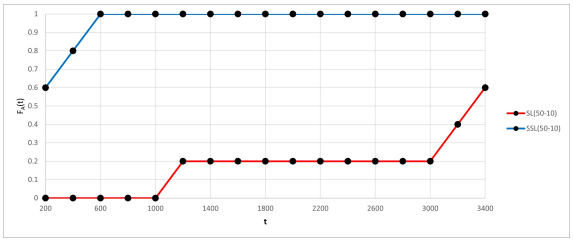
On the other hand, the impact of the weight combinations on the CPLEX performance clearly emerges. In particular, the weight combination (50 - 50) seems to make the instances more difficult to solve, as previously remarked. In fact, in case of SL the average optimality gap is higher than 4% within the stated time limit, whereas in case of SSL the running time increases on average of about the 10% with respect to the one related to the weight combination (10 - 10). Notice, however, that all the instances are solved to optimality in case of SSL regardless the selected weight combination. On the other hand, in case of SL this happens only by considering the weight combination (10 - 10).

To compare the performance of SL and SSL in a more accurate way, we also report the *performance profiles* of the two approaches with respect to solving time and optimality gap. As discussed in Dolan and Moré (2002), the performance profile for a solver is the (cumulative) distribution function of a performance measure. The comparison of performance profiles of different solvers may provide useful information about the relative performance of one solver against the others, often hidden when only comparing average results. Specifically, Figure 2 reports the performance profiles of SL and SSL with respect to solving time, separately for the four weight combinations discussed till now, while Figure 3 reports the performance profiles of SL and SSL, with respect to the percentage optimality gap, for the weight combinations (50-10), (10-50) and (50-50), by omitting those for (10-10) since for this setting all the instances are solved to optimality by both approaches (see Table 3). According to such performance profiles, the dominance of SSL over SL in terms of solving time and percentage optimality gap clearly

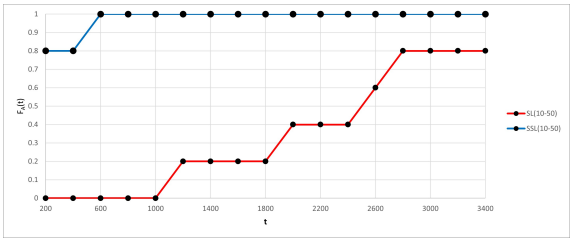
emerges for all weight combinations.



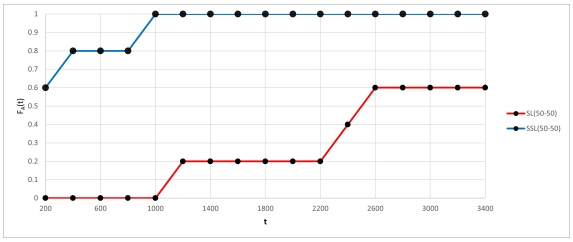
(a) Weight combination (10 - 10).



(b) Weight combination (50 - 10).

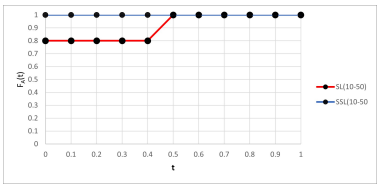


(c) Weight combination (10 - 50).

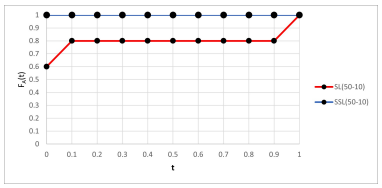


(d) Weight combination (50 - 50).

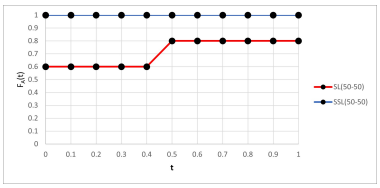
Figure 2: Performance profiles of SL (in red) and SSL (in blue) with respect to solving time.



(a) Weight combination (10 - 50).



(b) Weight combination (50 - 10).



(c) Weight combination (50 - 50).

Figure 3: Performance profiles of SL (in red) and SSL (in blue) with respect to percentage optimality gap.

Let us compare now the features of the solutions obtained by solving SL and SSL with CPLEX. According to Table 3, almost all the reported average indicators are pretty similar for SL and SSL, showing that the more aggregated super-storage location representation well approximates the original representation, based on single storage locations. The only relevant exception concerns the outgoing items, which in case of SSL are moved towards the collection area at a lower frequency with respect to SL (at this regard, compare the percentage saturation of the collection area after three hours for SL and SSL). This is the negative counterpart of having considered super-stacks rather than single stacks, tied with the security constraints (5) and (6) and the priority requirements expressed by (19) and (23). As an example, consider the case in which a given amount of items needs to be retrieved from two contiguous stacks, with two columns in the first stack. In case of SL, picking operations can be performed from the two contiguous stacks in the same period of time. In fact, the first stack can be completely emptied by a FKL, whereas another FKL can retrieve items from the contiguous stack immediately after, by respecting both the security constraints (the vehicles move on different arcs) and the priority requirements (the first stack is emptied before the second one). As opposed, since in SSL the two contiguous stacks are represented by a unique super-stack, only one vehicle can travel from the super-stack to the collection area during one period of time. Therefore, the second trip can be performed only when the first one is already concluded. The super-stack

representation may thus slightly slow down the frequency of the anticipation moves. However, the number of columns in the collection area at the end of the time horizon is the same for both SL and SSL (the collection area is in fact completely saturated in both cases after four hours), thus testifying that the SSL modelling issue does not affect seriously the anticipation moves. We can conclude that, at least on this set of artificial instances, the quality of the solutions found with SSL is not deteriorated with respect to the quality of the ones found with SL, with a considerable gain, however, in terms of required computational time.

6.4.2 Matheuristic approach evaluation

In the second set of experiments, we have solved the five artificial instances by means of the proposed matheuristic with the weight combination (10 - 10), which is the only one under which both SL and SSL always determine optimal solutions, and we have compared the computed solutions to the optimal ones in terms of computational time and percentage optimality gap. As described in Section 5, the matheuristic consists of a decomposition of the time horizon (i.e., the duration of a shift, which is four hours for the set of artificial instances) into smaller periods, i.e., into subshifts. The time length of a subshift is a key parameter of the approach. We tested two different time lengths, by splitting the time horizon into four and eight subshifts, thus obtaining subshifts of about 60 and 30 minutes, respectively. In the following, we shall refer to the resulting versions of the matheuristic by means of the notation NS-(total number of subshifts). The resolution of each subproblem was performed via CPLEX, by stopping the execution as soon as an optimality gap of 5% was reached. However, most subproblems were solved to optimality.

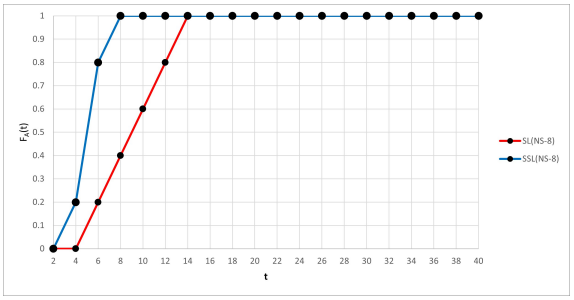
For each artificial instance, Table 4 reports, when using SL and SSL, the time (in seconds) required by CPLEX to find an optimal solution, the solving time of the matheuristic (calculated as the sum of the times needed to solve each subproblem) and the corresponding percentage optimality gap, by considering both time decompositions mentioned before.

When the SL formulation is used, the matheuristic seems to generate still difficult subproblems that CPLEX is hardly able to solve, despite the reduction in problem size led by the time horizon decomposition. In particular, NS-8 seems to be more affected by the performed time decomposition. In fact, although NS-8 is able to solve to optimality instance number 2, the average optimality gap of the computed solutions is about 19%, whereas it is around 11% in case of NS-4. However, its computational time is much shorter than the time required by NS-4. Overall, the average reduction in computational time is 99% for NS-8 and 97% for NS-4 compared to the time required by CPLEX.

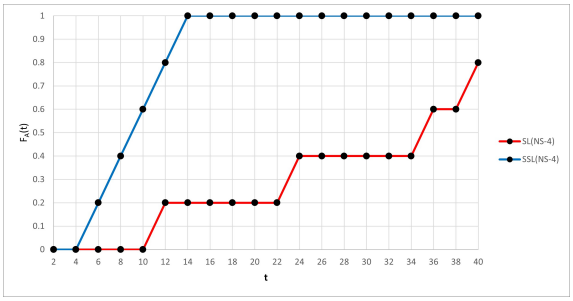
Better results are obtained when using formulation SSL: the average optimality gap is about 8% in case of NS-4 and 10% in case of NS-8, which seems again to be more affected by the performed time decomposition. Moreover, the time required by the matheuristic is on average the 95% lower than the one required by CPLEX for solving formulation SSL.

For a deeper comparison, in Figure 4 we report the performance profiles of the matheuristic with SL and SSL, separately for the two tested decomposition strategies, with respect to solving time (Figure 4a and Figure 4b) and optimality gap (Figure 4c and Figure 4d). Such profiles show the very good performance of the matheuristic with SSL for both solving time and optimality gap. Interestingly, in case of NS-4 the matheuristic with SL is able to solve one instance out of five with a lower optimality gap. Finally, in Figure 5 we show the performance profiles of the matheuristic with SSL, separately for NS-4 and NS-8, with respect to solving time (Figure 5a) and optimality gap (Figure 5b). If, on one side, the dominance of the time decomposition NS-8 clearly appears in terms of solving time, the gap profiles show instead a better performance only for a subset of the solved instances, by highlighting that NS-4 may be preferable in terms

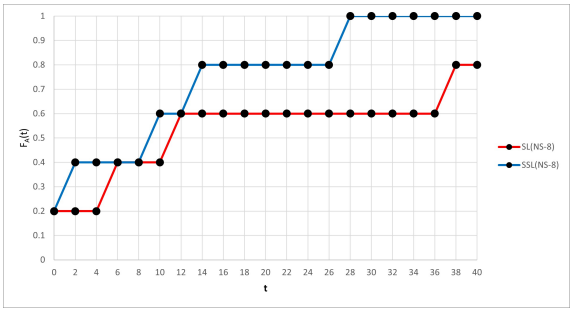
of optimality gap.



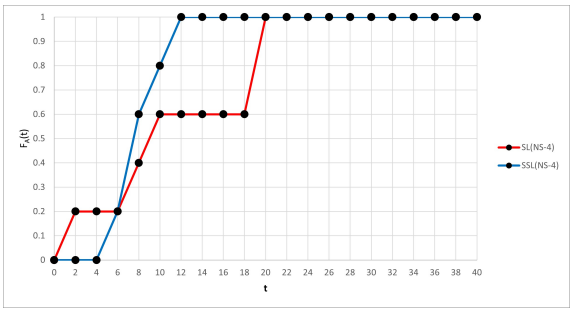
(a) NS-8: performance profiles of the matheuristic with SL and SSL with respect to solving time.



(b) NS-4: performance profiles of the matheuristic with SL and SSL with respect to solving time.



(c) NS-8: performance profiles of the matheuristic with SL and SSL with respect to percentage optimality gap.



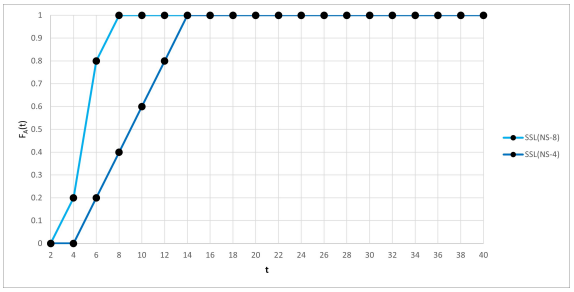
(d) NS-4: performance profiles of the matheuristic with SL and SSL with respect to percentage optimality gap.

Figure 4: Performance profiles of the matheuristic with SL (red) and SSL (blue) with respect to solving time and percentage optimality gap.

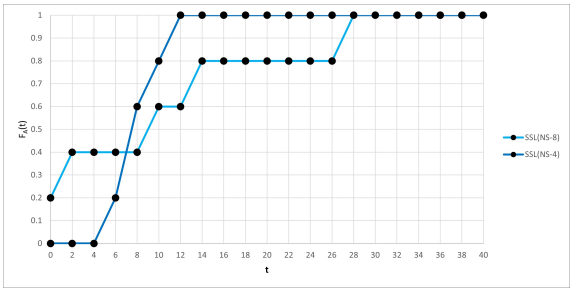
In conclusion, the proposed matheuristic appears to be very efficient and well suitable to address large scale real instances with both SL and SSL.

6.4.3 Managerial insights

In the third set of experiments, we performed an analysis aimed at obtaining some managerial insights on what makes the addressed SRP hard to solve. Specifically, we considered three relaxations of formulations SL and SSL with the weight combination (10 - 10), i.e., the only combination under which both SL and SSL always found optimal solutions. For each relaxation, we solved the artificial instances via CPLEX. The results are reported in Table 5 in case of SL and in Table 6 in case of SSL. In these tables, for each artificial instance, the time (in seconds) required by CPLEX to solve each relaxation is shown. The computational time (in seconds) needed to solve the complete SL or SSL formulation is also reported in the first row of Table 5 and Table 6, respectively. More in detail, in row "Priority-relax" of both tables we report the results for a problem relaxation where the storage and the retrieval policy constraints (18)-(23) are deleted, while row "Sec-relax" refers to a problem relaxation where the security constraints (5) and (6) are removed. Row "No routing restrictions" reports instead the results for a version of the problem without routing restrictions, i.e., where the vehicle routing constraints (3)–(6) are defined without restrictions on the areas of the warehouse where the vehicles can move. We also tested the impact of reducing the number of columns to move and of enlarging the addressed time horizon. The related results are reported in the last two rows of Table 5 and Table 6, which refer to the solution of formulations SL and SSL, respectively, when the number



(a) Performance profiles of the matheuristic with SSL, for NS-4 and NS-8, with respect to solving time.



(b) Performance profiles of the matheuristic with SSL, for NS-4 and NS-8, with respect to percentage optimality gap.

Figure 5: Performance profiles of the matheuristic with SSL with respect to solving time and percentage optimality gap.

of columns to move is reduced of about the 10% (row "10% reduced demand") and when the time horizon is extended of two hours (row "6 hours time horizon").

According to the results in Table 5 and comparing the three relaxations, it seems that what makes the problem really hard to solve are the storage and the retrieval policy constraints (18)-(23). By relaxing such constraints, in fact, the average time required by CPLEX reduces of about the 95% with respect to the one needed to solve the complete version of SL. Relaxing the security constraints or the routing restrictions, instead, does not seem to have a well defined impact on the difficulty of the resolution process, at least on this set of artificial instances. Sometimes, in fact, the resolution process is accelerated (see instance 2 in row "Sec-relax" and instance 1 in row "No routing restrictions"), whereas in other cases it appears to be more complicated. This could be explained by an increased number of feasible solutions, which CPLEX is not able to efficiently explore within the one hour time limit imposed. A decrease of the average running time required by CPLEX can be instead observed when the total amount of columns to move is reduced of the 10%. In this case, in fact, the average running time reduces of the 30%. Finally, when the time horizon is extended of two additional hours, the increased number of variables and constraints generate instances which are too hard to solve. In particular, only one optimal solution is found in this scenario. A similar trend can be observed by considering formulation SSL. According to the results in Table 6, in fact, it seems that relaxing storage and retrieval policy constraints as well as reducing the total amount of columns to move simplifies the resolution process. In both cases, in fact, the running time is reduced of about the 60% on average with respect to the one needed to solve SSL (see rows "Priority-relax" and "10% reduced demand"). However, regarding the security constraints and the routing restrictions, SSL seems to greatly benefit from their relaxation, and a decrease of the running time is indeed observed for the majority of the instances except, in both cases, for instance 3. Finally, when extending the time horizon, an optimal solution is always determined by solving SSL, despite the increased number of variables and constraints.

6.5 Tests on real instances

As outlined before, the matheuristic approach relies either on formulation SL or SSL, which are both characterized by parameters ψ and ξ, defining the mutual priorities between the secondary goals as well as their priorities with respect to the primary goal. An additional key parameter of the approach is the time length in which each shift is decomposed in order to define subshifts.

Regarding ψ and ξ, we tested the four weight combinations introduced in Section 6.4, i.e.,

(10-10), (10-50), (50-10) and (50-50). Recalling that a shift lasts eight hours in real scenarios, we tested three different time lengths for a subshift. Specifically, we considered the decomposition of the time horizon into 10 and 16 subshifts, which corresponds to have subshifts of about 60 and 30 minutes, as in the tests on the artificial instances reported in Section 6.4. Given the difficulty in solving real instances, we also considered a finer decomposition by splitting the time horizon into 30 subshifts, which corresponds to have subshifts of about 15 minutes. The tested shift decompositions are denoted with NS-10, NS-16 and NS-30, respectively.

For both SL and SSL, the three time lengths for a subshift and the four weight combinations of ψ and ξ have been combined. Each of the 15 real instances has thus been solved 12 times by the matheuristic based on SL (180 runs) and 12 times by the one based on SSL (180 runs), for a total of 360 runs.

Since the time limit required by the company to obtain solutions for an entire shift is 240 minutes, we imposed a different time limit on the resolution of the subproblems corresponding to the subshifts depending on whether 10, 16 or 30 subshifts are generated. Specifically, in case of 10 subproblems, the time limit per subproblem is 24 minutes; in case of 16 subproblems, the time limit per subproblem is 15 minutes; finally, in case of 30 subproblems, the time limit per subproblem is 8 minutes. In any case, the algorithm may stop the resolution of a subproblem before reaching the time limit, if the percentage gap between the optimum and the current solution value is lower than 10%.

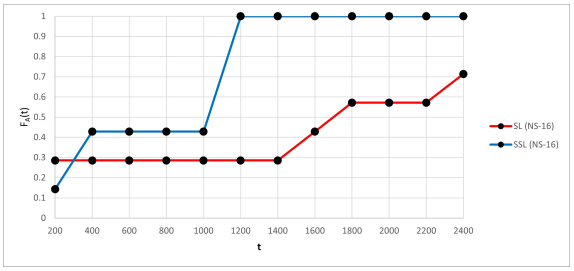
We firstly investigated the impact of the used formulation on the efficiency and efficacy of the matheuristic by comparing the total number of instances for which it is able to compute a solution within the time limit imposed. These numbers are reported in Table 7 for the alternative formulations SL and SSL, the three time lengths for subshifts, and the four combinations of ψ and ξ. We emphasize that, on this set of real instances, CPLEX was not able to determine feasible solutions or lower bounds when addressing the complete formulations SL and SSL. The same behavior was observed when considering the problem relaxations analysed in Section 6.4 for the artificial instance set.

Despite the reduction in problem size led by the proposed time horizon decomposition, when SL is considered, the matheuristic generates too large subproblems that CPLEX is hardly able to solve. CPLEX in fact finds solutions only to the minority of the tested instances. The finer time horizon decomposition, namely NS-30, seems to be the most suitable algorithm setting in this case, even though not all the 15 instances are successfully solved. Interestingly, the weight combination $(10 - 50)$ seems to generate the hardest subproblems. This may be explained by considering that, in any instance, the number of items requiring movements towards storage locations is lower than the number of items to move to the collection area. The latter, in fact, is associated with the present and the future demand to satisfy, due to the anticipation movements policy considered. Giving priority to outgoing movements may thus generate much more busy scenarios within the system (e.g., more busy collectors, or not availability of FKL to move incoming items towards stacks), which are harder to face within the time limit imposed.

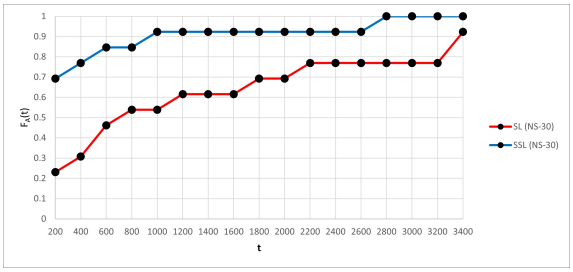
The SSL formulation, instead, seems to be much more effective in addressing the problem, in accordance with the trend observed for the artificial instance set, being able to solve 131 out of 180 runs. Notice that it successfully solves all the instances when both NS-16 and NS-30 are coupled with the (10-10), (50-10) and (50-50) weight combinations, thus suggesting that SSL, the time decomposition given by NS-16 and NS-30, and the weight combinations (10-10), (50-10) and (50-50) are appropriate settings for the efficiency of the proposed resolution approach. The option NS-10, instead, seems to be not suitable for generating subproblems that CPLEX can easily address within the given time limit. Moreover, as for SL, the weight combination $(10 - 50)$ appears to generate too hard subproblems. Therefore, this weight combination will be no longer discussed.

To have additional insights on the performance of the matheuristic when using either SL or SSL, in Table 8 we report the average solving time and some aggregated features of the computed solutions in terms of crucial performance indicators suggested by our industrial partner. Results refer to NS-16 and NS-30, by limiting the discussion to the weight combination (10-10). Precisely, in case of NS-16, the results refer to the subset of 7 real instances which are solved by the matheuristic when using both SL and SSL, whereas in case of NS-30, the results refer to the subset of 13 real instances which are solved by both versions of the matheuristic (according to Table 7).

Specifically, the primary goal is analysed in terms of the average LGV and FKL travelling time, expressed in minutes. Regarding the secondary goals, i.e., the emptying of conveyors and the anticipation movements towards the collection area, the first one is measured in terms of the average time, in minutes, incoming items idle on conveyors before be moved on an available collector. On the other hand, the second one is measured in terms of the average saturation level of the collection area. Specifically, two measures are reported: the average time, in minutes, in which the collection area is completely saturated (Collection area Saturation 100%), and the average time, always in minutes, in which the collection area is full at least at its 90% (Collection area Saturation $\geq 90\%$).



(a) NS-16: performance profiles of the matheuristic with SL and SSL with respect to solving time.



(b) NS-30: performance profiles of the matheuristic with SL and SSL with respect to solving time.

Figure 6: Performance profiles of the matheuristic with SL (red) and SSL (blue) with respect to solving time.

Table 8 shows the difficulty of the matheuristic in solving the real instances when using SL and NS-16. In this case, in fact, the subproblem resolution normally stopped because the time limit was reached, thus providing fairly optimized solutions. Moreover, the average travel time of both LGV and FKL when using SL is about 15% higher than the one when using SSL. A lower average idle time on conveyors is remarkable (look at the indicator Conveyors Avg. Idle Time per column), at the expense however of a worst exploitation of the collection area. Similar results can be observed when considering NS-30 and comparing SL and SSL. Indeed, the average travelling time is pretty similar for both LGV and FKL but, when using SL, the matheuristic outperforms in terms of average idle time on conveyors, while when using SSL it is able to better exploit the collection area. In any case, the time saving when using SSL rather than SL is remarkable, being about 75% lower.

To better compare SL and SSL in terms of solving time, in Figure 6 we also report the performance profiles of the matheuristic with SL and SSL, separately for NS-16 and NS-30, with respect to this performance metric (Figure 6a and Figure 6b). The profiles have been computed over the 7 and the 13 instances solved for the two time decompositions, respectively, by both approaches. According to the profiles, the dominance of SSL over SL clearly emerges also for the real instances. In case of NS-16, however, the matheuristic with SL is able to solve one instance over seven in less time.

Focusing on the matheuristic based on SSL, which proved to be more efficient according to the previously presented results, Table 9 reports the same indicators in Table 8 by considering NS-16 and NS-30, and the weight combinations (10-10), (50-10) and (50-50). With these choices, in fact, the matheuristic was able to solve all the 15 real instances (see Table 7).

The version NS-30 seems to be faster in finding solutions with respect to NS-16, which however is still under the time limit imposed. Nevertheless, NS-30 is not able to optimize the travel time of the fleet of vehicles as good as NS-16 does, worsening the solutions of 18% for the travel time of the fleet of the LGV, and of 12% for the travel time of the fleet of the FKL with respect to NS-16 (on average over the three parameter settings). This may be explained by considering that increasing the number of subshifts surely defines smaller, and thus easier, subproblems to tackle, but at the same time may make the model myopic of the near future. This is confirmed also looking at the indicator Conveyors Avg. Idle Time, i.e., the average time of permanence of an incoming item on a conveyor. The results related to the exploitation of the collection area are quite similar for NS-16 and NS-30, with NS-30 slightly outperforming NS-16 in terms of the time the collection area is completely full. However, being the latter only a secondary goal and coming at the expenses of a high increase of travel times for NS-30, NS-16 seems to address a more suitable time horizon splitting for the set of the real instances. Therefore, it is the only one discussed next. Regarding the weight combinations for NS-16, by increasing weight ψ from 10 to 50 and keeping $\xi = 10$, the idle time of incoming items on conveyors decreases, as expected, at the expense though of an increase of the average LGV and FKL travel times. Finally, the weight combination (10-10) outperforms the weight combination (50-50) in all the reported primary goal indicators. Moreover, by comparing the average solving times of NS-16 with weight combinations (10-10) and (50-50), the latter appears to generate more tricky subproblems. Therefore, only NS-16 with the weight combinations (10-10) and (50-10) is further discussed.

Table 10 and Table 11 report other features of the solutions obtained by considering SSL, NS-16, and the weight combinations (10-10) and (50-10). Specifically, Table 10 shows the minimum, the maximum and the average time (in minutes) each vehicle has travelled over the 15 instances. Standard deviation is also reported. Table 11 shows instead the average time (in minutes) columns idle on the collection area before been loaded on trucks, the percentage of items being picked from their storage locations and directly moved to the collection area with no stop on collectors, the average time items spend idling on a collector separately for products in K_{in} and in K_{out}, and finally the average time the collectors are full at least at their 60%. The latter is calculated as the average time in minutes all the six collectors are filled with a number of items exceeding the above mentioned saturation level over the 15 instances.

As outlined in Table 10, travel times for the same type of vehicles seem to be quite balanced on average for both types of weight combinations. Moreover, according to Table 11, prioritizing the emptying of conveyors, i.e., increasing ψ from 10 to 50, not only causes a decrease of idle time of incoming items on conveyors, as observed before, but also a decrease of idle time of incoming items on collectors. Incoming items are thus faster moved from the conveyors towards their assigned stacks when the weight combination (50-10) is chosen.

Prioritizing the conveyors emptying movements does also affect the movements of outgoing items. In fact, the number of outgoing items being retrieved from their stacks and directly transported to the collection area is slightly increased, implying a lower exploitation of collectors by outgoing items as well as a decrease of the average time outgoing items spend idling on collectors. Also observe that the average idle time of outgoing items in the collection area decreases of about the 7% when the weight combination (50-10) is chosen. This may be explained by considering that, when using the weight combination (50-10), the movements towards the collection area are delayed in order to prioritize the movements of incoming freight from col-

lectors to stacks performed by FKL. Outgoing items are thus retrieved from stacks later than when the weight combination (10-10) is chosen, idling less time in the collection area.

Finally, note that the average idle time of incoming and outgoing items on collectors is very low when considering both weight combinations, and that the saturation of collectors exceeds the 60% of their capacities for only a few minutes on average, thus testifying a very good synchronization among vehicles for the movements of items, so avoiding congestion on collectors.

By summarizing, decomposing each shift into 16 subshifts of equal length, and solving the resulting subproblems via formulation SSL, under either the setting (10-10) or the setting (50-10) for parameters ψ and ξ, appears to be an efficient algorithmic strategy to solve the addressed SRP on real scenarios, by obtaining solutions of good quality in terms of travel times of the vehicles and their synchronization, and also in terms of an effective exploitation of collectors and collection area within the warehouse.

6.6 Worst-case scenario analysis

For the worst-case scenario analysis, we have considered one of the busiest weeks for the company with respect to both production and shipments, just before a peak period of requests. Indeed, in the selected week both production and shipments are higher of about the 25% with respect to a normal week, and about 500 more movements are required for storing or retrieving items per shift. Days are solved in cascade, from the first shift of the first day of the week till the last one. We considered formulation SSL, and we used the option NS-16, i.e., we split each shift into 16 subshifts, and the weight combination $\psi = 10$ and $\xi = 10$. The main motivation for considering the weight combination (10-10) is that, working on a weekly basis and focusing on a week with a very high rotation index, both storing and retrieving operations appear to be particularly crucial to manage, and therefore any sort of prioritization might bring to too expensive results in terms of algorithm solving time.

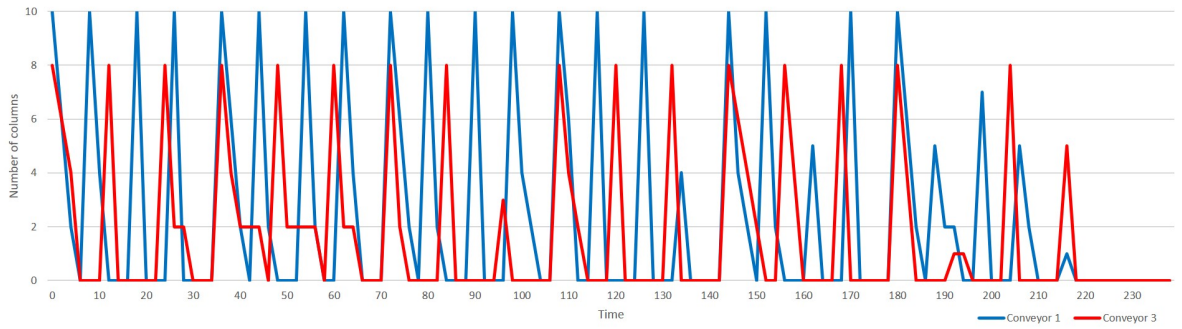
Under the considered setting, the matheuristic we propose is able to determine a solution to all the shifts composing the week under study. Table 12 reports the same kinds of results reported in Table 9 and Table 11. In particular, the first column refers to the busy week under study, the second column summarizes the results already reported in Table 9 and Table 11 for option NS-16 and the weight combination $\psi = 10$ and $\xi = 10$, which refer to an ordinary number of operations within the warehouse, and the third column shows the difference in percentage between the the first two columns.

The increased number of movements requested in the selected busy week causes an unavoidable increase of travel times for both LGV and FKL (+18% and +22%, respectively). Conveyors are strongly used in this busy week and, being releases more frequent than in ordinary periods, they are required to be emptied by LGV in a faster way not to block the production of the site (recall that production decisions are independent of warehouse management). Indeed, the idle time on conveyors of incoming items is slightly decreased with respect to the more ordinary shifts (of about the 4%). Similarly, the average idle time of incoming items on collectors is decreased (of about the 27%). Therefore, faster movements of incoming items from conveyors to stacks are performed in this busy week with respect to more ordinary weeks.

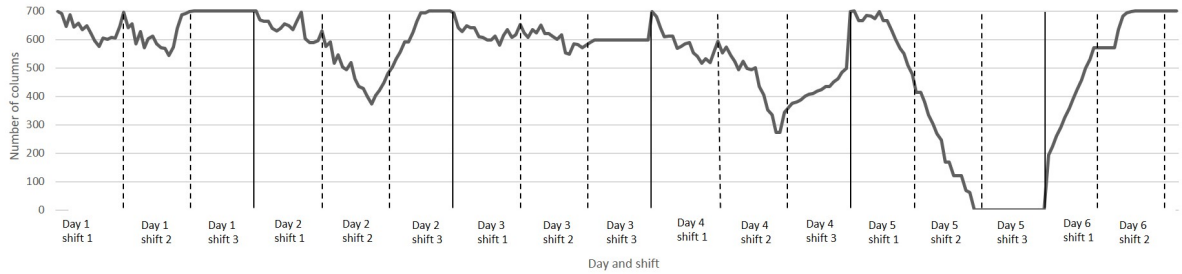
Direct movements of outgoing items from stacks to the collection area are increased in this busy week (compare the indicator "Qty directly to Collection area"). Nevertheless, for those outgoing items passing through a collector on their itinerary towards the collection area, a longer (+31%) idle time on collectors is observable. Additionally, the collection area is saturated for less time (compare the indicators "Collection area Saturation 100% and $\geq$ 90%" for both scenarios), but anticipation movements are performed in advance, as testified by the longer (+3%) idle time

of columns in the collection area.

Figure 7a and Figure 7b report the saturation trends of two crucial spots of the warehouse for specific periods. Figure 7a shows the number of columns released and idling on conveyor 1



(a) Saturation level of conveyors 1 and 3 in shift 1 of day 1.



(b) Saturation of the collection area.

Figure 7: Saturation trends of conveyors and collection area.

and conveyor 3 during the first shift of day 1 of the considered week. Notice that the statistics for conveyor 2 is not reported as the amount of released items is almost 0, thus conveyors 1 and 3 are extremely exploited. The selected day has both a production rate and a shipment request higher than the average calculated over all the shifts of the week. The unit of measure of the time reported on the x-axis is four minutes. The capacity of conveyor 1 and conveyor 3 is 10 and 8 columns, respectively. The LGV empty conveyors with large advance with respect to new releases, thus avoiding production delays caused by busy conveyors. Only a small amount of items remains idling for a long time, which is however less than 30 minutes. Figure 7b reports instead the number of columns idling in the collection area during the entire week (the last shift of the last day is not reported as the saturation has been already reached during the previous shift). In general, the collection area is well exploited. Especially during the third shift of each day (i.e., the night shift), a very high number of columns are moved towards the collection area. Recall that, during this shift, production still continues and storage operations are required, so workers are not dedicated to replenishment only. This behaviour is evident during the third shift of days 2 and 4. At the end of day 5, the collection area is completely emptied, since no shipments are planned on day 6. However, on day 6 the shipping list of the first day of the next week is available and replenishment of the collection area can start again. Saturation is reached during the second shift of that day.

Finally, regarding the average solving time required by the approach to solve a shift of the worst-case week under consideration, it is much lower than the time the company requires to solve a shift, that is four hours (see Table 12). The proposed matheuristic appears thus to be a valuable tool for solving the considered SRP problem also in real worst-case scenarios.

7 Conclusions

This paper discusses a sequencing and routing problem originated from a real-world application context in tissue logistics. Specifically, the problem consists in defining the best sequence of locations to visit within a warehouse for the storage and/or retrieval of a given set of items during a specified time horizon, by considering some additional requirements. In particular, an anticipation movement policy and a strict order of precedence to fill and retrieve items in/from storage locations have to be considered when planning the operations of two fleets of different types of vehicles, having movements restrictions within the warehouse. The first policy is pursued due to the high number of movements daily requested, to anticipate operations with respect to peak and very busy periods. On the other hand, the order of precedence is pursued due to the perishability of the products managed within the warehouse.

We have modelled the problem as a constrained multicommodity flow problem on an space-time network, and we have proposed two Mixed-Integer Linear Programming formulations as well as a matheuristic approach based on the decomposition of the time horizon. Precisely, the original problem has been split into subproblems that can be easily addressed via a state-of-the-art optimization solver, and solved in cascade. A wide experimental analysis has been presented on real instances provided by our industrial partner, showing the efficiency and the efficacy of the proposed matheuristic approach. Moreover, since the presented formulations could not be solved on the real instances due to their size, tests on a set of smaller artificial instances, based however on real data, have been conducted as well, in order to provide some comparative evaluation of the achieved results and outline some useful managerial insights.

We plan to generalize the achieved results along three main directions of research: i) studying a combined optimization problem which integrates picking and put-away operations with assignment storage location decisions; the assignment of storage locations, the scheduling of put-away and picking operations, and the routing of the vehicles inside the warehouse define in fact hard interdependent decisions which are very challenging to address; ii) taking into account different warehouse layouts and storage/retrieval policies, in even larger systems with a higher number of vehicles; iii) considering a green version of the studied SRP, where some vehicles are conventional, i.e., with internal combustion engine, while others are electric, i.e., equipped with a lithium-ion battery. Furthermore, another interesting avenue of research which we plan to pursue is to devise special SRP cases which are solvable in polynomial time. To the best of the authors' knowledge, in fact, polynomially solvable cases have not been studied so far.

Appendix A

This appendix is devoted to specify the set of nodes and arcs composing the subgraphs $\mathcal{G}_{in} = (\mathcal{N}_{in}, \mathcal{A}_{in})$, $\mathcal{G}_{out} = (\mathcal{N}_{out}, \mathcal{A}_{out})$, $\mathcal{G}_{F1} = (\mathcal{N}_{F1}, \mathcal{A}_{F1})$ and $\mathcal{G}_{F2} = (\mathcal{N}_{F2}, \mathcal{A}_{F2})$, introduced in Section 4.1, where commodities $k \in \mathcal{K}_{in}$ and $k \in \mathcal{K}_{out}$, and vehicles $v \in \mathcal{V}^1$ and $v \in \mathcal{V}^2$ may move, respectively.

Incoming freight flow of product type $k \in \mathcal{K}_{in}$ is originated from an input point $r \in \mathcal{R}$, it passes through some collectors $b \in \mathcal{B}$, it may possibly pass through some storage locations in $\mathcal{S}_{out}$, and some storage locations $s^{k'} \in \mathcal{S}^{k'}_{in}$, with $k' \in \mathcal{K}_{in} \setminus \{k\}$, finally reaching its assigned storage location $s^k \in \mathcal{S}^k_{in}$. Thus, the set of nodes $\mathcal{N}_{in}$ is defined as follows:

$$\mathcal{N}_{in} := \{(i,t) \in \mathcal{N} : \ i \in \mathcal{R} \cup \mathcal{B} \cup \mathcal{S}_{out} \cup \mathcal{S}_{in}\}.$$

The set of movement arcs for incoming freight flow of product type $k \in \mathcal{K}_{in}$ is composed of arcs defined as follows:

- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{R},\ j \in \mathcal{B}$;
- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{R},\ j \in \mathcal{R},\ i \neq j$;
- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{B},\ j \in \mathcal{S}_{out} \cup \mathcal{S}_{in}$;
- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{B},\ j \in \mathcal{B},\ i \neq j$;
- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{S}_{out} \cup \mathcal{S}_{in},\ j \in \mathcal{S}_{out} \cup \mathcal{S}_{in},\ i \neq j$.

The set of holding arcs for incoming freight flow of product type $k \in \mathcal{K}_{in}$ is composed of arcs defined as follows:

$$\{((i,t),(i,t+1)) \in \mathcal{A}:\ (i,t) \in \mathcal{N}_{in},\ (i,t+1) \in \mathcal{N}_{in}\}.$$

Outgoing freight flow of product type $k \in \mathcal{K}_{out}$ is originated from a storage location $s^k \in \mathcal{S}^k_{out}$, it may possibly pass through some storage locations $s^{k'} \in \mathcal{S}^{k'}_{out}$, with $k' \in \mathcal{K}_{out} \setminus \{k\}$, and some storage locations in $\mathcal{S}_{in}$, the collectors $b \in \mathcal{B}$, finally reaching the collection area π. Thus, the set of nodes $\mathcal{N}_{out}$ is defined as follows:

$$\mathcal{N}_{out} := \{(i,t) \in \mathcal{N}:\ i \in \mathcal{S}_{out} \cup \mathcal{S}_{in} \cup \mathcal{B} \cup \{\pi\}\}.$$

The set of movement arcs for outgoing freight flow of product type $k \in \mathcal{K}_{out}$ is composed of arcs defined as follows:

- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{S}_{out} \cup \mathcal{S}_{in},\ j = \pi$;
- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{S}_{out} \cup S_{in},\ j \in \mathcal{B}$;
- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{S}_{out} \cup \mathcal{S}_{in},\ j \in \mathcal{S}_{out} \cup \mathcal{S}_{in},\ i \neq j$;
- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{B},\ j = \pi$;
- $((i,t),(j,t')) \in \mathcal{A}:\ i \in \mathcal{B},\ j \in \mathcal{B},\ i \neq j$.

The set of holding arcs for outgoing freight flow of product type $k \in \mathcal{K}_{out}$ is composed of arcs defined as follows:

$$\{((i,t),(i,t+1)) \in \mathcal{A}:\ (i,t) \in \mathcal{N}_{out},\ (i,t+1) \in \mathcal{N}_{out}\}.$$

A vehicle of type $v \in \mathcal{V}^1$ may only move in the hallway, between the input points $r \in \mathcal{R}$, the collectors $b \in \mathcal{B}$ and the parking area ω^1. Thus, the set of nodes $\mathcal{N}_{F1}$ is defined as follows:

$$\mathcal{N}_{F1} := \left\{(i,t) \in \mathcal{N}:\ i \in \mathcal{R} \cup \mathcal{B} \cup \left\{\omega^1\right\}\right\}.$$

The set of movement arcs of vehicle $v \in \mathcal{V}^1$ is composed of arcs defined as follows:

- $((i,t),(j,t')) \in \mathcal{A}: i = \omega^1,\ j \in \mathcal{R}$;
- $((i,t),(j,t')) \in \mathcal{A}: i \in \mathcal{R},\ j \in \mathcal{B}$;
- $((i,t),(j,t')) \in \mathcal{A}: i \in \mathcal{R},\ j \in \mathcal{R},\ i \neq j$;
- $((i,t),(j,t')) \in \mathcal{A}: i \in \mathcal{B},\ j \in \mathcal{R}$;
- $((i,t),(j,t')) \in \mathcal{A}: i \in \mathcal{B},\ j = \omega^1$;

- $((i,t),(j,t')) \in \mathcal{A} : i \in \mathcal{B},\ j \in \mathcal{B},\ i \neq j.$

The set of holding arcs of vehicle $v \in \mathcal{V}^1$ is composed of arcs defined as follows:

$$\{((i,t),(i,t+1)) \in \mathcal{A} :\ (i,t) \in \mathcal{N}_{F1},\ (i,t+1) \in \mathcal{N}_{F1}\}.$$

A vehicle of type $v \in \mathcal{V}^2$ may only move in the storage area, between the collectors $b \in \mathcal{B}$, the sets of storage locations $\mathcal{S}_{out}$ and $\mathcal{S}_{in}$, the collection area π and the parking area ω^2. Thus, the set of nodes $\mathcal{N}_{F2}$ is defined as follows:

$$\mathcal{N}_{F2} := \left\{(i,t) \in \mathcal{N} :\ i \in \mathcal{B} \cup \mathcal{S}_{out} \cup \mathcal{S}_{in} \cup \{\pi\} \cup \left\{\omega^2\right\}\right\}.$$

The set of movement arcs of vehicle $v \in \mathcal{V}^2$ is composed of arcs defined as follows:

- $((i,t),(j,t')) \in \mathcal{A} :\ i = \omega^2,\ j \in \mathcal{S}_{out}$;
- $((i,t),(j,t')) \in \mathcal{A} :\ i = \omega^2,\ j \in \mathcal{B}$;
- $((i,t),(j,t')) \in \mathcal{A} :\ i \in \mathcal{S}_{in} \cup \mathcal{B},\ j = \omega^2$;
- $((i,t),(j,t')) \in \mathcal{A} :\ i \in \mathcal{S}_{out} \cup \mathcal{B},\ j = \pi$;
- $((i,t),(j,t')) \in \mathcal{A} :\ i \in \mathcal{B},\ j \in \mathcal{S}_{out} \cup \mathcal{S}_{in}$;
- $((i,t),(j,t')) \in \mathcal{A} :\ i \in \mathcal{B},\ j \in \mathcal{B},\ i \neq j$;
- $((i,t),(j,t')) \in \mathcal{A} :\ i \in \mathcal{S}_{out} \cup \mathcal{S}_{in},\ j \in \mathcal{B}$;
- $((i,t),(j,t')) \in \mathcal{A} :\ i \in \mathcal{S}_{out} \cup \mathcal{S}_{in},\ j \in \mathcal{S}_{out} \cup \mathcal{S}_{in},\ i \neq j$;
- $((i,t),(j,t')) \in \mathcal{A} :\ i = \pi,\ j = \omega^2$.

The set of holding arcs of vehicle $v \in \mathcal{V}^2$ is composed of arcs defined as follows:

$$\{((i,t),(i,t+1)) \in \mathcal{A} :\ (i,t) \in \mathcal{N}_{F2},\ (i,t+1) \in \mathcal{N}_{F2}\}.$$

Acknowledgments

The authors thank the Associate Editor and the three anonymous referees for the interesting comments and suggestions, which allowed to greatly improve the previous versions of the paper. The research has been supported by Region of Tuscany-Regional Government (POR FESR 2014-2020-Line 1-Research and Development Strategic Projects), through the Project IREAD4.0 under Grant CUP 7165.24052017.112000028, and by Polo Universitario Sistemi Logistici di Livorno.

References

Ballestín, F., Pérez, Á., Lino, P., Quintanilla, S., and Valls, V. (2013). Static and dynamic policies with rfid for the scheduling of retrieval and storage warehouse operations. *Computers & Industrial Engineering*, 66(4):696–709.

Ballestín, F., Pérez, Á., and Quintanilla, S. (2020). A multistage heuristic for storage and retrieval problems in a warehouse with random storage. *International Transactions in Operational Research*, 27(3):1699–1728.

Boysen, N., Briskorn, D., and Emde, S. (2017). Sequencing of picking orders in mobile rack warehouses. *European Journal of Operational Research*, 259(1):293–307.

Briant, O., Cambazard, H., Cattaruzza, D., Catusse, N., Ladier, A.-L., and Ogier, M. (2020). An efficient and general approach for the joint order batching and picker routing problem. *European Journal of Operational Research*.

Carlo, H. J., Vis, I. F., and Roodbergen, K. J. (2014). Storage yard operations in container terminals: Literature overview, trends, and research directions. *European Journal of Operational Research*, 235(2):412–430.

Chabot, T., Lahyani, R., Coelho, L. C., and Renaud, J. (2017). Order picking problems under weight, fragility and category constraints. *International Journal of Production Research*, 55(21):6361–6379.

Chen, F., Wang, H., Qi, C., and Xie, Y. (2013). An ant colony optimization routing algorithm for two order pickers with congestion consideration. *Computers & Industrial Engineering*, 66(1):77–85.

Chen, F., Wang, H., Xie, Y., and Qi, C. (2016). An aco-based online routing method for multiple order pickers with congestion consideration in warehouse. *Journal of Intelligent Manufacturing*, 27(2):389–408.

Cinar, D., Oliveira, J. A., Topcu, Y. I., and Pardalos, P. M. (2017). Scheduling the truckload operations in automated warehouses with alternative aisles for pallets. *Applied Soft Computing*, 52:566–574.

Cortés, P., Gómez-Montoya, R. A., Muñuzuri, J., and Correa-Espinal, A. (2017). A tabu search approach to solving the picking routing problem for large-and medium-size distribution centres considering the availability of inventory and k heterogeneous material handling equipment. *Applied Soft Computing*, 53:61–73.

Crainic, T. G., Ricciardi, N., and Storchi, G. (2009). Models for evaluating and planning city logistics systems. *Transportation Science*, 43(4):432–454.

Cuda, R., Guastaroba, G., and Speranza, M. G. (2015). A survey on two-echelon routing problems. *Computers & Operations Research*, 55:185–199.

Davarzani, H. and Norrman, A. (2015). Toward a relevant agenda for warehousing research: literature review and practitioners' input. *Logistics Research*, 8(1):1.

de Brito, M. P. and De Koster, M. (2004). Product and material returns: handling and warehousing issues. In *Reverse Logistics*, pages 135–153. Springer.

De Koster, R., Le-Duc, T., and Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, 182(2):481–501.

De Santis, R., Montanari, R., Vignali, G., and Bottani, E. (2018). An adapted ant colony optimization algorithm for the minimization of the travel distance of pickers in manual warehouses. *European Journal of Operational Research*, 267(1):120–137.

Dolan, E. D. and Moré, J. J. (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213.

Gagliardi, J.-P., Renaud, J., and Ruiz, A. (2012). Models for automated storage and retrieval systems: a literature review. *International Journal of Production Research*, 50(24):7110–7125.

Gómez-Montoya, R. A., Cano, J. A., Cortés, P., and Salazar, F. (2020). A discrete particle swarm optimization to solve the put-away routing problem in distribution centres. *Computation*, 8(4):99.

Gong, Y. and De Koster, R. B. (2011). A review on stochastic models and analysis of warehouse operations. *Logistics Research*, 3(4):191–205.

Grosse, E. H., Glock, C. H., Jaber, M. Y., and Neumann, W. P. (2015). Incorporating human factors in order picking planning models: framework and research opportunities. *International Journal of Production Research*, 53(3):695–717.

Gu, J., Goetschalckx, M., and McGinnis, L. F. (2007). Research on warehouse operation: A comprehensive review. *European Journal of Operational Research*, 177(1):1–21.

Gue, K. R., Ivanović, G., and Meller, R. D. (2012). A unit-load warehouse with multiple pickup and deposit points and non-traditional aisles. *Transportation Research Part E: Logistics and Transportation Review*, 48(4):795–806.

Hemmelmayr, V. C., Cordeau, J.-F., and Crainic, T. G. (2012). An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers & Operations Research*, 39(12):3215–3228.

Küçükyaşar, M., Y. Ekren, B., and Lerher, T. (2020). Cost and performance comparison for tier-captive and tier-to-tier sbs/rs warehouse configurations. *International Transactions in Operational Research*.

Lanza, G., Passacantando, M., and Scutellà, M. G. (2022). Assigning and sequencing storage locations under a two level storage policy: Optimization model and matheuristic approaches. *Omega*, 108:102565.

Lu, W., McFarlane, D., Giannikas, V., and Zhang, Q. (2016). An algorithm for dynamic order-picking in warehouse operations. *European Journal of Operational Research*, 248(1):107–122.

Masae, M., Glock, C. H., and Grosse, E. H. (2020). Order picker routing in warehouses: A systematic literature review. *International Journal of Production Economics*, 224:107564.

Matusiak, M., de Koster, R., Kroon, L., and Saarinen, J. (2014). A fast simulated annealing method for batching precedence-constrained customer orders in a warehouse. *European Journal of Operational Research*, 236(3):968–977.

Mowrey, C. H. and Parikh, P. J. (2014). Mixed-width aisle configurations for order picking in distribution centers. *European Journal of Operational Research*, 232(1):87–97.

Oliveira, J. A. (2007). Scheduling the truckload operations in automatic warehouses. *European Journal of Operational Research*, 179(3):723–735.

Pan, J. C.-H. and Wu, M.-H. (2012). Throughput analysis for order picking system with multiple pickers and aisle congestion considerations. *Computers & Operations Research*, 39(7):1661–1672.

Pohl, L. M., Meller, R. D., and Gue, K. R. (2009a). An analysis of dual-command operations in common warehouse designs. *Transportation Research Part E: Logistics and Transportation Review*, 45(3):367–379.

Pohl, L. M., Meller, R. D., and Gue, K. R. (2009b). Optimizing fishbone aisles for dual-command operations in a warehouse. *Naval Research Logistics*, 56(5):389–403.

Rouwenhorst, B., Reuter, B., Stockrahm, V., van Houtum, G.-J., Mantel, R., and Zijm, W. H. (2000). Warehouse design and control: Framework and literature review. *European Journal of Operational Research*, 122(3):515–533.

Scholz, A., Henn, S., Stuhlmann, M., and Wäscher, G. (2016). A new mathematical programming formulation for the single-picker routing problem. *European Journal of Operational Research*, 253(1):68–84.

Schrotenboer, A. H., Wruck, S., Roodbergen, K. J., Veenstra, M., and Dijkstra, A. S. (2017). Order picker routing with product returns and interaction delays. *International Journal of Production Research*, 55(21):6394–6406.

Tappia, E., Roy, D., Melacini, M., and De Koster, R. (2019). Integrated storage-order picking systems: Technology, performance models, and design insights. *European Journal of Operational Research*, 274(3):947–965.

van Gils, T., Ramaekers, K., Caris, A., and de Koster, R. B. (2018). Designing efficient order picking systems by combining planning problems: State-of-the-art classification and review. *European Journal of Operational Research*, 267(1):1–15.

Wang, Y., Liu, Z., Huang, K., Mou, S., and Zhang, R. (2020). Model and solution approaches for retrieval operations in a multi-tier shuttle warehouse system. *Computers & Industrial Engineering*, 141:106283.

Weidinger, F., Boysen, N., and Schneider, M. (2019). Picker routing in the mixed-shelves warehouses of e-commerce retailers. *European Journal of Operational Research*, 274(2):501–515.

Wruck, S., Vis, I. F., and Boter, J. (2013). Time-restricted batching models and solution approaches for integrated forward and return product flow handling in warehouses. *Journal of the Operational Research Society*, 64(10):1505–1516.

Zhao, X., Zhang, R., Zhang, N., Wang, Y., Jin, M., and Mou, S. (2020). Analysis of the shuttle-based storage and retrieval system. *IEEE Access*, 8:146154–146165.

Žulj, I., Glock, C. H., Grosse, E. H., and Schneider, M. (2018). Picker routing and storage-assignment strategies for precedence-constrained order picking. *Computers & Industrial Engineering*, 123:338–347.

Table 1: Sets, parameters and variables used in the models.

Sets	
T	no. time instants in which the time horizon is discretized
$\tilde{T}$	no. time instants for anticipation of movements
$\mathcal{K}_{in}$	set of incoming product types
$\mathcal{K}_{out}$	set of outgoing product types
$\mathcal{V}^1$	set of vehicle of fleet $F1$
$\mathcal{V}^2$	set of vehicle of fleet $F2$
ω^1, ω^2	parking areas for vehicles of $F1$ and $F2$
$\mathcal{R}$	set of input points (e.g., conveyors)
$\mathcal{B}$	set of collectors
π	collection area
$\mathcal{S}_{in}^k$	set of storage locations assigned to product type $k \in \mathcal{K}_{in}$
$\mathcal{S}_{out}^k$	set of storage locations occupied by product type $k \in \mathcal{K}_{out}$
$\mathcal{S}^k$	set of storage locations occupied/assigned to product type $k \in \mathcal{K}$
$\tilde{\mathcal{S}}_{in}^k$	set of super-storage locations assigned to product type $k \in \mathcal{K}_{in}$
$\tilde{\mathcal{S}}_{out}^k$	set of super-storage locations occupied by product type $k \in \mathcal{K}_{out}$
$\mathcal{G}_{in} = (\mathcal{N}_{in}, \mathcal{A}_{in})$	subgraph where product type $k \in \mathcal{K}_{in}$ may move
$\mathcal{G}_{out} = (\mathcal{N}_{out}, \mathcal{A}_{out})$	subgraph where product type $k \in \mathcal{K}_{out}$ may move
$\mathcal{G}_{F1} = (\mathcal{N}_{F1}, \mathcal{A}_{F1})$	subgraph where vehicle $v \in \mathcal{V}^1$ may move
$\mathcal{G}_{F2} = (\mathcal{N}_{F2}, \mathcal{A}_{F2})$	subgraph where vehicle $v \in \mathcal{V}^2$ may move
Parameters	
$d_{in}^k(r,t)$	no. items of product type $k \in \mathcal{K}_{in}$ released on $r \in \mathcal{R}$ at time t
$d_{out}^k(\pi,t)$	no. items of product type $k \in \mathcal{K}_{out}$ requested in π at time t
u_r^k	no. items of product type $k \in \mathcal{K}$ positioned on $r \in \mathcal{R}$ at $t = 0$
u_b^k	no. items of product type $k \in \mathcal{K}$ positioned on $b \in \mathcal{B}$ at $t = 0$
u_π^k	no. items of product type $k \in \mathcal{K}$ positioned in π at $t = 0$
c_s	capacity of storage location $s \in \mathcal{S}_{in} \cup \mathcal{S}_{out}$
$\tilde{c}_{\tilde{s}}$	capacity of super-storage location $\tilde{s} \in \tilde{\mathcal{S}}_{in} \cup \tilde{S}_{out}$
c_r	capacity of $r \in \mathcal{R}$
c_π	capacity of π
c_b	capacity of $b \in \mathcal{B}$
c_{F1}, c_{F2}	capacity of $v \in \mathcal{V}^1$ or $v \in \mathcal{V}^2$
$\tau_{i,j}$	travel time between location i and j within the warehouse
Variables	
$x_{(i,t)(j,t')}^v \in \{0,1\}$	model the routing of vehicles $v \in \mathcal{V}$
$y_{(i,t)(j,t')}^k \in \mathbb{Z}_+$	model the itinerary of items of product type $k \in \mathcal{K}$
$\alpha(s^k,t) \in \{0,1\}$	model the sequencing policy for $s^k \in \mathcal{S}_{in}^k$
$\beta(s^k,t) \in \{0,1\}$	model the sequencing policy for $s^k \in \mathcal{S}_{out}^k$

Table 2: Features of artificial and real instances.

Artificial Instances

ID	stacks	super-stacks	K_{in}	K_{out}	C_{in}	C_{out}
1	16	10	3	6	142	288
2	14	4	2	4	108	234
3	18	9	3	5	78	188
4	13	4	2	3	132	226
5	15	8	3	5	134	364
Average	13.4	6.6	2.2	4.2	90.8	117.2

Real Instances

ID	stacks	super-stacks	K_{in}	K_{out}	C_{in}	C_{out}
1	108	55	8	42	335	1009
2	81	48	8	35	233	963
3	51	42	13	29	422	629
4	101	46	9	38	58	981
5	83	43	9	32	368	1014
6	66	43	11	28	335	174
7	105	43	7	27	233	1318
8	71	35	7	11	422	666
9	28	27	8	0	48	0
10	83	35	4	12	368	541
11	56	32	4	20	350	1120
12	8	6	6	0	64	0
13	165	73	10	36	434	1639
14	115	68	10	43	368	1692
15	68	50	12	41	378	1113
Average	82.4	45.8	8.4	28	318.8	1112.8

Table 3: Performance measures and features of solutions using SL and SSL.

	SL			
(ψ - ξ)	(10 - 10)	(50 - 10)	(10 - 50)	(50 - 50)
Avg. Optimality Gap %	0%	0.20%	0.10%	4.30%
Avg. Solving Time (sec.)	2237	2945	2364	2601
LGV Avg. Travel Time (min.)	75.68	75.44	75.68	75.52
FKL Avg. Travel Time (min.)	119.71	119.77	119.66	122.29
Conveyors Avg. Idle Time per column (min.)	0.113	0.112	0.113	0.112
% of saturation of collection area after 3h	96.92%	94.78%	95.14%	96.80%
% of saturation of collection area after 4h	100%	100%	100%	100%
	SSL			
(ψ - ξ)	(10 - 10)	(50 - 10)	(10 - 50)	(50 - 50)
Avg. Optimality Gap %	0%	0%	0%	0%
Avg. Solving Time (sec.)	217	166	115	240
LGV Avg. Travel Time (min.)	75.28	75.28	75.28	75.20
FKL Avg. Travel Time (min.)	120.00	119.94	120.00	120.00
Conveyors Avg. Idle Time per column (min.)	0.113	0.110	0.113	0.113
% of saturation of collection area after 3h	91.45%	90.50%	91.57%	90.38%
% of saturation of collection area after 4h	100%	100%	100%	100%

Table 4: Performance of the matheuristic approach with SL and SSL.

	SL					SSL				
	CPLEX	NS-8		NS-4		CPLEX	NS-8		NS-4	
Inst.	Time	Time	Gap%	Time	Gap%	Time	Time	Gap%	Time	Gap%
1	2612	8	10.29%	35	9.44%	607	6	8.87%	11	8.39%
2	2053	6	0.00%	12	6.18%	13	3	0.00%	5	6.22%
3	1061	12	41.81%	40	18.39%	53	5	13.85%	10	7.33%
4	2938	9	4.49%	23	0.34%	17	5	1.78%	8	5.10%
5	2520	11	36.80%	148	19.71%	396	7	26.92%	13	11.66%
Avg.	2237	9	18.68%	52	10.81%	217	5	10.28%	10	7.74%

Table 5: SL: tests on relaxations, column reduction and time horizon extension.

	Instance				
Version	1	2	3	4	5
SL	2612.01	2053.43	1061.50	2938.08	2519.82
Priority-relax	35.32	49.04	86.44	118.94	53.62
Sec-relax	3600.00	1869.37	3600.00	3600.00	3600.00
No routing restrictions	2425.32	3600.00	3600.00	3600.00	3600.00
10% reduced demand	1885.84	541.61	859.98	2088.82	2242.56
6 hours time horizon	820.71	3600.00	3600.00	3600.00	3600.00

Table 6: SSL: tests on relaxations, column reduction and time horizon extension.

	Instance				
Version	1	2	3	4	5
SSL	606.57	12.70	53.40	16.53	395.84
Priority-relax	33.58	9.46	15.54	11.45	19.45
Sec-relax	198.07	11.33	325.70	11.48	339.39
No routing restrictions	172.63	11.90	127.98	14.84	78.67
10% reduced demand	23.42	11.45	12.50	10.25	19.93
6 hours time horizon	175.85	15.40	394.90	15.28	69.21

Table 7: Number of instances solved by the matheuristic approach.

Weights		SL			SSL		
ψ	ξ	NS-10	NS-16	NS-30	NS-10	NS-16	NS-30
10	10	3	7	13	6	15	15
50	10	3	7	13	6	15	15
10	50	1	3	4	6	7	9
50	50	3	6	12	7	15	15

Table 8: Matheuristic solution comparison (SL vs SSL).

	NS-16[1]		NS-30[2]	
$\psi = 10, \quad \xi = 10$	SL	SSL	SL	SSL
Avg. Solving Time (sec.)	2431	589	1637	401
LGV Avg. Travel Time (min.)	164.80	143.83	281.88	284.92
FKL Avg. Travel Time (min.)	231.10	195.22	296.11	296.92
Conveyors Avg. Idle Time per column (min.)	1.93	2.01	3.47	4.88
Collection area Saturation 100% (min.)	26.57	55.14	18.46	41.54
Collection area Saturation $\geq 90\%$ (min.)	76.29	153.14	71.69	148.31

[1] Averages over 7 instances solved by both SL and SSL.
[2] Averages over 13 instances solved by both SL and SSL.

Table 9: Features of solutions (SSL, options NS-16 and NS-30).

	NS-16		
(ψ - ξ)	(10 - 10)	(50 - 10)	(50 - 50)
Avg. Solving Time (sec.)	1673	1488	2408
LGV Avg. Travel Time (min.)	234.29	237.89	248.59
FKL Avg. Travel Time (min.)	254.69	274.00	266.11
Conveyors Avg. Idle Time per column (min.)	1.74	1.52	1.92
Collection area Saturation 100% (min.)	58.53	50.13	63.73
Collection area Saturation $\geq$ 90% (min.)	172.00	149.33	208.27
	NS-30		
(ψ - ξ)	(10 - 10)	(50 - 10)	(50 - 50)
Avg. Solving Time (sec.)	471	403	676
LGV Avg. Travel Time (min.)	276.72	293.15	306.40
FKL Avg. Travel Time (min.)	289.12	302.76	312.04
Conveyors Avg. Idle Time per column (min.)	4.40	1.83	4.18
Collection area Saturation 100% (min.)	64.93	59.60	64.27
Collection area Saturation $\geq$ 90% (min.)	166.53	156.53	172.40

Table 10: Travel time details for the fleet of vehicles (in minutes).

	NS-16, $\psi = 10$, $\xi = 10$				NS-16, $\psi = 50$, $\xi = 10$			
LGV	Min.	Max.	Avg.	Std. Dev.	Min.	Max.	Avg.	Std. Dev.
1	38	448	242	113.8	42	400	236	101.7
2	42	416	232	113.3	40	434	235	107.6
3	58	418	233	108.7	54	424	233	105.0
4	50	430	234	107.4	48	422	244	104.0
5	58	426	230	106.5	54	410	242	95.7
FKL	Min.	Max.	Avg.	Std. Dev.	Min.	Max.	Avg.	Std. Dev.
1	40	474	265	128.5	34	476	282	131.0
2	32	476	248	132.6	42	476	265	134.4
3	0	480	248	139.9	0	476	271	136.9
4	34	480	247	141.6	40	468	272	127.2
5	26	480	257	135.9	24	480	271	137.9
6	30	480	255	135.4	28	470	280	129.7
7	36	470	261	133.0	26	480	277	127.7

Table 11: Collection area and collectors details.

(ψ - ξ)	NS-16 (10 - 10)	NS-16 (50 - 10)
Avg. Idle Time in Collection area per column (min.)	386	357
Qty directly to Collection area	92%	93%
Avg. Idle Time on Collectors per column K_{in} (min.)	1.50	1.20
Avg. Idle Time on Collectors per column K_{out} (min.)	7.2	5.1
Saturation of Collectors $\geq 60\%$ (min.)	20	20

Table 12: Features of solutions for NS-16 in a worst-case scenario.

NS-16, $\psi = 10$, $\xi = 10$			
	Busy	Ordinary	% Diff.
Avg. Solving Time (sec.)	2556	1673	+53%
LGV Avg. Travel Time (min.)	275.87	234.29	+18%
FKL Avg. Travel Time (min.)	309.60	254.69	+22%
Conveyor Avg. Idle Time (min.)	1.67	1.74	−4%
Collection area Saturation 100% (min.)	46	60	−23%
Collection area Saturation $\geq 90\%$ (min.)	144	172	−16%
Avg. Idle Time in Collection area (min.)	399	386	+3%
Qty directly to Collection area	96%	92%	+4%
Avg. Idle Time on Collectors per column K_{in} (min.)	1.1	1.5	−27%
Avg. Idle Time on Collectors per column K_{out} (min.)	9.4	7.2	+31%
Saturation of Collectors $\geq 60\%$ (min.)	18	20	−10%