Optimizing Two-Tier Robotized Sorting Systems for Urban Parcel Delivery

Junsu Kim[a,b], **Reem Khir**[a]

[a] Edwardson School of Industrial Engineering, Purdue University, West Lafayette, IN

[b] Graduate School of Logistics, Inha University, Incheon, Republic of Korea

jskim0305@inha.edu, rkhir@purdue.edu

Abstract

This paper addresses an operational planning challenge in two-tier robotized sorting systems (T-RSS), an emerging alternative to traditional conveyor-based sorting in e-commerce delivery stations. Designed to be compact and space-efficient, T-RSS use an upper tier to sort parcels from loading stations to drop-off points, which connect to roll containers on a lower tier where parcels are consolidated and transported to shipping docks for outbound trucks. While this architecture supports fast parcel processing, it also tightly couples the two tiers: parcel sorting decisions directly determine roll-container flows and downstream movement to outbound docks, creating a need for integrated planning across sorting and outbound operations. We address this challenge by developing a path-based sort planning model that *jointly* determines parcel-to-loading station, parcel-to-drop-off point, roll container-to-shipping dock, and truck-to-shipping dock assignments to efficiently route parcels through the hub. To enable scalability, we propose an *offline–then–online* solution framework that replaces parcel-level planning with an aggregated, lower-fidelity commodity-flow model in the *offline* phase, and then uses the resulting sorting paths *online* to guide real-time parcel routing. Results of computational experiments demonstrate the value of decision integration by comparing fully and partially integrated planning schemes across tiers, and offer practical insights on how to best exploit the operational potential of T-RSS in last-mile delivery stations.

***K*eywords** Automated Sorting Operations · Last-mile Delivery · Optimization · Facility Logistics

1 Introduction

The rapid growth of online retail has increased pressure on logistics networks to manage rising parcel volumes with greater speed and efficiency. This trend, coupled with workforce shortages and rising labor costs, sparked growing interest in advanced automation solutions (Szeszák et al., 2025). Among the various logistical tasks, sorting operations have become indispensable within the complex e-commerce supply chains, and increasingly automated, to enhance accuracy and efficiency while ensuring reliable and timely order fulfillment (Boysen et al., 2019). The primary function of a sorting system is to organize and consolidate parcel flows for outbound shipping, typically grouping parcels by features such as destination, service level, or order number to improve economies of scale in transportation. These systems vary in their level of automation, ranging from manual and semi-automated to fully automated sorters.

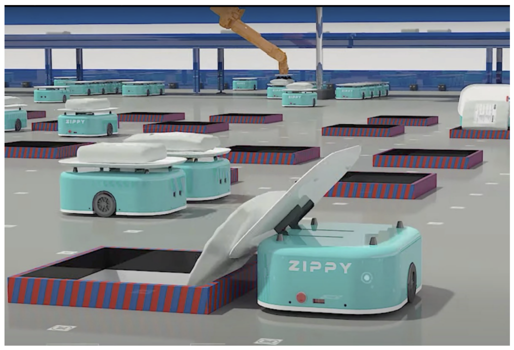
Traditional automated systems primarily rely on conveyors, and have proven effective for high-volume sorting operations, such as those operated in regional and inter-city distribution centers (Chen et al., 2024a). However, their bulky, rigid hardware configurations limit both scalability and adaptability to fluctuating demand (Boysen et al., 2017). In response, retailers have started adopting robotic sorting systems powered by autonomous mobile robots (AMRs), shifting away from fixed infrastructure toward more flexible and compact designs. For example, since 2019 Amazon has deployed Pegasus robots in some of its delivery stations to automate the handling of small, sortable parcels arriving from fulfillment and sorting centers, thereby improving sorting efficiency and service responsiveness (Amazon, 2022, 2025a). Likewise, Kmart Australia has introduced the tSort system by Tompkins Robotics at its distribution centers, sustaining throughput while reducing spatial requirements for logistics operations by 60% (Robotics, 2024).

Robotic sorting systems (RSS) have gained traction as effective solutions for last-mile sorting, representing the final stage of material handling before parcels are delivered to customers. While traditional parcel carriers like UPS and FedEx have seen limited use of RSS so far, the growing demand for last-mile sorting centers in the post-pandemic era may encourage wider adoption in various distribution and e-commerce contexts. For instance, Target, a major American retail corporation, has opened and continues to expand a network of sorting facilities that process online orders by receiving packages from nearby stores and subsequently sorting and consolidating them for local delivery partners (Target, 2024). These facilities decouple parcel sorting from store-based fulfillment, aiming to accelerate last-mile deliveries in metropolitan areas. Such urban facilities are expected to increasingly adopt robot-based sorting technologies in place of traditional conveyors, driven by space constraints and the demand for operational flexibility (Zhao et al., 2024). This development has also prompted the design of robotic systems with multi-level layouts that require careful coordination and may benefit from integrated decision-making across tiers (Tadumadze et al., 2023). Building on this development, we examine a closely related problem, which is described in detail below.

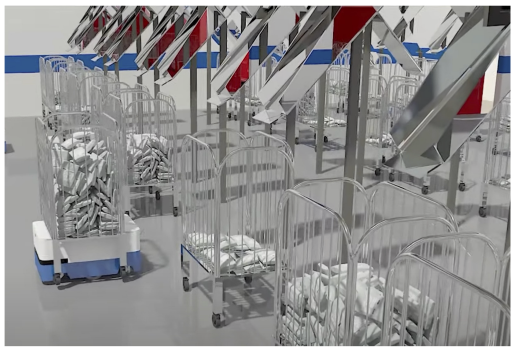
1.1 Motivating example

This paper investigates a parcel sorting process for last-mile delivery, addressing the operational planning challenges of two-tier robotized sorting systems (T-RSS). An example setting is depicted in Figure 1.

The sorting process in T-RSS begins with sort induction, where parcels are moved from the inbound area to the upper tier's sorting area via conveyors. The upper tier comprises multiple loading stations and drop-off points. At each loading station, parcels are transferred from the conveyor to the AMRs using automated mechanical pick arms. The AMRs then navigate the aisles to deliver parcels to their designated drop-off points. These drop-off points connect to roll containers on the lower tier through spiral conveyors, with each roll container assigned to a specific drop-off point. Once full, the roll containers—consolidating parcels for the same destination—are transported by mobile robots to shipping docks in the same lower tier, where delivery trucks await. Trucks are expected to depart once their assigned parcel requirements



(a) Snapshot of the upper tier of the T-RSS



(b) Snapshot of the lower tier of the T-RSS

Figure 1: Two-tier robotized sorting systems (source: Zippy sorting robot for intelligent sorting from Addverb, used with permission).

are met, initiating delivery to customers along pre-determined routes. Figure 2 schematically illustrates the four core assignment decisions in T-RSS, distributed across the first and second tiers. The upper tier determines the assignment of parcels to loading stations and drop-off points, while the lower tier governs the allocation of roll containers and outbound trucks to shipping docks.

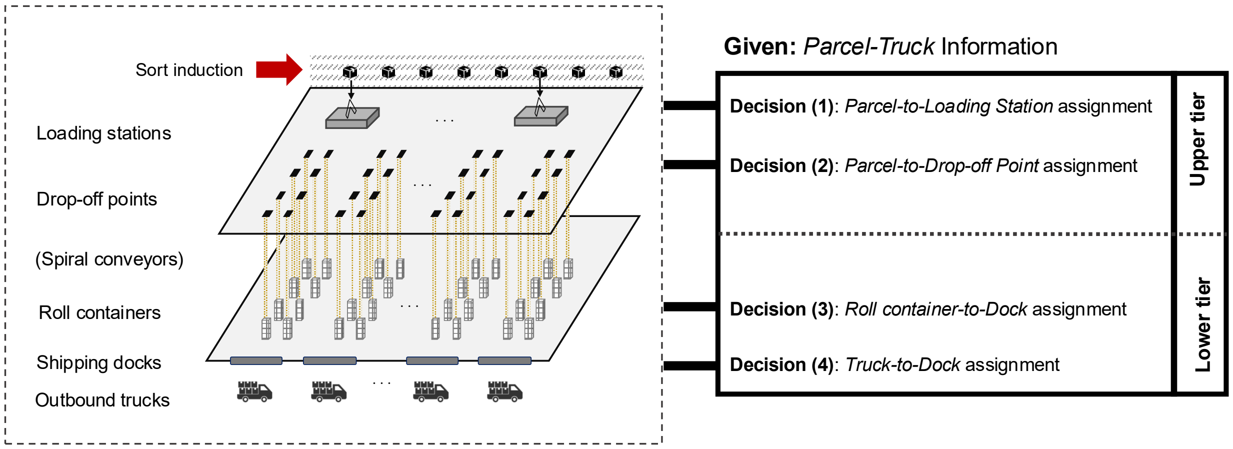

Sort induction
Loading stations
Drop-off points
(Spiral conveyors)
Roll containers
Shipping docks
Outbound trucks
Given: Parcel-Truck Information
Decision (1): Parcel-to-Loading Station assignment
Decision (2): Parcel-to-Drop-off Point assignment
Decision (3): Roll container-to-Dock assignment
Decision (4): Truck-to-Dock assignment
Upper tier
Lower tier

Figure 2: Schematic representation of two-tier robotized sorting systems and assignment decisions.

Although the sorting process ultimately concludes in the lower tier with truck loading, prior studies have concentrated predominantly on upper-tier operations, leaving the downstream configuration unaddressed. For example, Zou et al. (2021) developed closed queuing network models to estimate throughput capacity under different layouts, path topologies, and robot-to-loading station assignment policies. Their focus was on performance estimation and operating rules at the design and tactical levels, rather than on modeling explicit interactions across tiers. Xu et al. (2022) examined parcel-to-loading station assignment using an open queuing network approximation and a tabu search algorithm, balancing parcel waiting times and robot travel times. Similarly, Chen et al. (2024b) investigated the assignment of parcel destinations to drop-off points, explicitly modeling robot congestion through an open queuing network and addressing the trade-off between congestion and travel distance. More recently, Fang et al. (2025) introduced dynamic robot routing and destination reassignment policies, combining a semi-open queuing network with a Markov decision process framework to mitigate congestion under time-varying demand.

These studies share two notable characteristics. First, they primarily address *parcel-level assignment* problems within the upper tier, motivated by the central role of parcel-to-loading station or parcel-to-drop-off assignment in determining system throughput. Second, even when dynamic congestion and workload balancing are considered for more responsive robot control, the scope remains confined to the *upper-tier* sorting process, treating the eventual flow from roll containers to outbound trucks as an exogenous matter. This perspective aligns with traditional conveyor-based sorters, where outbound decisions after sorting are typically treated as tactical and relatively static, adjusted only in response to major demand shifts (Alpan et al., 2011; Jarrah et al., 2016).

By contrast, the flexibility of T-RSS fundamentally alters this hierarchy. The coupling between upper- and lower-tier decisions implies that even truck-to-dock and roll-container-to-dock assignments–often perceived as external to the sorting system–may affect throughput, and optimizing them jointly with parcel-level decisions can yield system-wide benefits. Overlooking these decisions may risk under-utilizing the adaptive potential of T-RSS, particularly in urban delivery contexts where parcel demand fluctuates on a daily or even intra-day basis.

1.2 Contribution

Building on these insights, this paper studies the joint optimization of upper- and lower-tier decisions in T-RSS, addressing a key gap in the existing literature. The primary contributions of this study are summarized as follows:

1. It presents a mixed-integer programming (MIP) formulation for the operational planning of T-RSS, explicitly capturing the often-overlooked impact of lower-tier configuration decisions alongside upper-tier sorting operations. Using the *actual* realized demand at the *individual* parcel level, the model generates parcel-level plans that optimize end-to-end sorting, from sort induction to truck loading, and provides a high-fidelity benchmark for evaluating more aggregated approaches.
2. It proposes a two-stage framework with *offline planning* and *online control* for practical deployment. In the offline stage, the *same* demand information is used in *aggregate* form (e.g., truck-level demand for the upcoming shift, known before parcels arrive at the hub) to optimize pre-planned parcel paths via either an aggregation-based relaxation of the parcel-level MIP or a solver-independent heuristic. In the online stage, as parcels arrive and are inducted, a control policy assigns them to these precomputed paths. We further establish a parcel–commodity equivalence that identifies when an aggregated solution can be lifted to a parcel-feasible plan without loss in objective value and specify a simple recovery procedure otherwise.
3. It presents extensive computational experiments over various layout sizes, demonstrating the efficiency and effectiveness of the proposed approaches in producing high-quality solutions. It also examines the *impact*

of decision integration across tiers by comparing fully and partially joint scenarios, revealing substantial operational gains from coordinated two-tier planning.

The remainder of the paper is structured as follows. Section 2 reviews the relevant literature on assignment decisions, traces the evolution of warehouse sorting systems, and identifies the research gap addressed in this study. Section 3 defines the problem and outlines the assumptions. Section 4 presents the optimization model for two-tier robotized parcel sorting processes. Section 5 extends the problem by introducing the solution framework. Section 6 discusses the computational results of our solution methods and explores the value of decision integration. Section 7 concludes the paper, highlighting the study's limitations and proposing directions for future research.

2 Literature Review

The optimization of consolidation and sorting operations, central to parcel hubs, ensure that parcels are accurately and efficiently dispatched to their intended destinations (Chen et al., 2024a). Despite their critical role in facility logistics, research on the operational planning of parcel sorting remains limited, especially compared to inbound or outbound truck scheduling and order-picking operations. It is worth noting that the evolution of sorting systems from manual to advanced robotic solutions highlights the logistics sector's adaptation to changing demands and technological advancements. Therefore, this section highlights several relevant works of this development pathway that mark significant advances in sorting operations from an assignment optimization perspective.

2.1 Optimizing parcel sorting in manual and conveyor-based systems

Early sorting operations relied heavily on manual labor. McAree et al. (2002) addressed assignment problems in a manual package sorting facility, focusing on optimally assigning inbound and outbound unit load devices to specific bins and racks. Their model aimed to minimize operational and capital costs through optimal forklift placement and movement strategies. To reduce the computational burden of their model, the authors proposed two heuristic algorithms that decomposed the problem into tractable sub-problems, ultimately achieving over 50% reduction in total transportation time. Werners and Wülfing (2010) studied a U-shaped layout at a Deutsche Post World Net parcel hub, focusing on the assignment of delivery groups to endpoints and loading gates. They incorporated chance constraints to improve robustness against volume fluctuations and achieved a 39% reduction in manual transport effort. While these studies did not involve automation, they laid important groundwork by showing how inter-dependent assignment decisions, *when jointly optimized*, can lead to significant efficiency gains.

The transition to conveyor-based automation has enabled higher throughput and greater precision, particularly in large regional parcel hubs. Jarrah et al. (2016) examined two linked assignment problems in a conveyor-based sorting center: (1) destination-to-door assignment, minimizing disruptive changes across shifts, and (2) loader-to-door assignment, balancing workloads. They proposed a hierarchical optimization model using pattern variables and practical constraints to sequentially prioritize these objectives. Notably, in such systems, these assignments represent tactical design decisions aimed at achieving good average performance, rather than being frequently reconfigured. This is because making changes can take a long time due to conveyor lineup reconfiguration requirements and can disrupt overall flow control. More recently, Boysen et al. (2024) formulated the loop sorter scheduling problem for closed-loop conveyor systems, integrating tray-to-order assignment, lane assignment, and sequencing. They proposed a tabu search heuristic that outperformed standard priority rules, emphasizing the value of decision integration. For a comprehensive overview of conveyor-based sorting operations, we refer the reader to Boysen et al. (2019).

Despite their high throughput, conveyor systems often lack flexibility to accommodate demand variability. To address this, several studies have explored semi-automated systems, which combine automated primary sorting with manual or flexible secondary sorting (Khir et al., 2021). Novoa et al. (2018) employed stochastic modeling to account for variability in parcel flows, producing sorting plans that balanced average performance and robustness. Similarly, Khir et al. (2023) developed robust optimization models that account for time and/or volume variability in two-stage sorting environments. Their approach introduced simplified constraints to maintain time-feasible operations while enhancing computational tractability.

2.2 Optimizing parcel sorting in robotized systems

This study focuses on the emergence of mobile robot-based parcel sorting, which offers greater flexibility and adaptability in parcel movement and sorting routes compared to conventional conveyor-based systems (Chen et al., 2024b; Zou et al., 2021). Although mobile robot-based systems have been extensively studied in the contexts of robotic mobile fulfillment systems and smart factory material handling, their application to parcel sorting has received limited attention. To date, only a few studies have investigated RSS, summarized in what follows.

Zou et al. (2021) were the first to explore RSS by analyzing layout configurations in both single-tier and two-tier systems and evaluating rule-based operational policies. The primary distinction between these systems lies in whether parcels are delivered directly to their final destination on a single level or first consolidated at a drop-off point before getting transferred to the lower tier for truck loading. Using closed queuing networks and simulation validated with data from Deppon Express (China), they evaluated system performance under different layout and path designs, focusing on robot-to-loading station assignment rules to enhance efficiency and reduce costs.

Building on this, Xu et al. (2022) introduced an open queuing network model to capture conveyor congestion effects, estimating system performance for parcel-to-loading station assignments in the upper tier. They also proposed a MIP model that minimizes order throughput time under station capacity constraints. In the same context, Chen et al. (2024b) addressed destination-to-drop-off point assignments, first using a queuing model to account for robot congestion and then developing an optimization model with a heuristic algorithm. However, these studies treat drop-off points in the upper tier as final destinations, without explicitly modeling the handover to lower-tier truck loading, thus limiting their scope to partial system planning. Moreover, their focus is on system design decisions rather than daily operational planning, which is the focus of our work.

A more dynamic perspective is offered by Fang et al. (2025), proposing a three-stage framework to manage robot routing and destination re-assignment in T-RSS. Their approach comprises a semi-open queuing network for performance estimation, a Markov decision process for dynamic routing, and a MIP for destination re-assignment. Since their primary objective is to develop solution algorithms for robot control software, their study is complementary to ours and contributes to the development of more comprehensive warehousing solutions. A notable feature of their approach is the grouping of six drop-off points into a zone, reflecting their use-case company's operational environment and aiming to balance congestion across zones.

Despite differing slightly in system context, Boysen et al. (2023) tackled a closely related problem of optimizing parcel sorting in robotized environments. They proposed a two-step multiple scenario approach, combining optimization and heuristics to address uncertainty in parcel sequences. By generating and solving deterministic scenarios and applying a consensus function to select final decisions, their approach is particularly suited for stochastic, real-time problems. However, their study is limited to single-tier systems operating a single loading station with a focus on piece-to-order assignment in fulfillment centers, where SKUs are picked and packed. In contrast, our study addresses an independent facility focused on sorting incoming, pre-packed parcels for last-mile delivery, where rapid consolidation and dispatch efficiency are critical. Table 1 summarizes the relevant literature on RSS and situates our work within this context.

Table 1: Summary of relevant studies on robotized sorting systems.

Reference	**Decision Level**		**System Structure**		**Operational Assignment Decisions**
	Layout Design	Operating Policies	One-Tier	Two-Tier (lower tier decisions?)	
Zou et al. (2021)	✓	✓	✓	✓ (No)	Robot-to-loading station
Xu et al. (2022)	✓	✓		✓ (No)	Parcel-to-loading station
Chen et al. (2024)	✓	✓		✓ (No)	Destination-to-drop-off point
Fang et al. (2025)		✓		✓ (No)	Robot routing; Destination-to-zone
Boysen et al. (2023)		✓	✓		Piece-to-order; Order-to-collection point; Piece-to-robot
This Study		✓		✓ (Yes)	Parcel-to-loading station; Parcel-to-drop-off point; Roll container-to-shipping dock; Truck-to-shipping dock

While integrated decision-making has been emphasized in manual and conveyor-based sorting systems, its impact remains largely unexplored for robotic systems. Our study addresses this gap by quantifying the benefits of integration across tiers and offering guidelines for system planning and control.

3 Problem Description

We focus on the operational planning of a fully automated facility employing T-RSS. The goal of this facility is to process standardized, small sortable shipping parcels intended for customers in urban areas, serving a role similar to that of a distribution or e-commerce delivery station.

Define a *sort wave* to represent an operational shift during which incoming parcels are sorted and loaded onto their designated delivery trucks. Let P be the set of incoming parcels during the sort wave, and K the set of available trucks during the same wave. Define $o_{p,k}$ as an indicator that equals 1 if parcel $p \in P$ can be loaded onto truck $k \in K$, and 0 otherwise. Each parcel is linked to exactly one truck, as it represents a box containing items from a single order. We presume that the volume of incoming parcels during each sort wave is sufficient to keep sorting resources engaged, while ensuring adequate resources to manage all parcels. An adequate number of loading stations, drop-off points, and roll containers are available to meet all destination requirements, following the principles outlined by Fedtke and Boysen (2017) and Tan et al. (2021).

Upper-tier operations Each parcel moves from the inbound area to the sort area via induction conveyor, where it first gets directed to a loading station. Define I to be the set of available loading stations where mechanical picking arms are placed to swiftly transfer parcels onto mobile robots. The robot then travel through the upper-tier aisles to reach drop-off points located in the middle area. Define J to be the set of drop-off points, where each point is connected to a roll container in the lower tier.

Two potential queueing issues may occur at the loading station: (1) parcels waiting to be loaded onto robots, and (2) robots waiting for parcels to arrive. To mitigate the first issue, we assume that a sufficient number of robots are available at the loading stations, similar to Chen et al. (2024b); Xu et al. (2022), and that advanced picking arm technology are employed for rapid parcel loading, thereby avoiding service capacity problems (Tan et al., 2021). The second issue, though possible, is negligible for sorting plan generation since the model does not determine which specific robots transport parcels. Additionally, the consideration of parcel inter-arrival times and travel times from the sort induction to the assigned loading station may compensate for any potential robot waiting time.

Lower-tier operations Parcel flows cascade into lower-tier operations, triggering the assignment of roll containers and destination trucks to shipping docks, denoted by D. Each roll container has a limited capacity τ and is assigned to a single dock, meaning it can only carry parcels destined for the truck assigned to that outbound dock. However, due to the limited capacity of each container, a single truck may be served by multiple containers.

Delivery trucks are assumed to arrive punctually and remain available throughout each sort wave, ensuring that loading is completed by the end of the shift. In this approach, drivers enter within the designated window and depart once loading is finalized, thereby reducing scheduling conflicts and overlaps (Boysen et al., 2021). In addition, each dock is assumed to accommodate at most one truck per sort wave, and therefore, the truck sequencing problem is not considered in this study.

The key decisions in the problem span both tiers of the system, and can be represented as a multi-partite graph to facilitate understanding of parcel flows, as depicted in Figure 3. Specifically, the **upper tier** involves (1) *parcel-to-loading station (P2L)* and (2) *parcel-to-drop-off point (P2DP)* assignments, while the **lower tier** involves (3) *roll container-to-shipping dock (C2D)* and (4) *truck-to-shipping dock (T2D)* assignments.

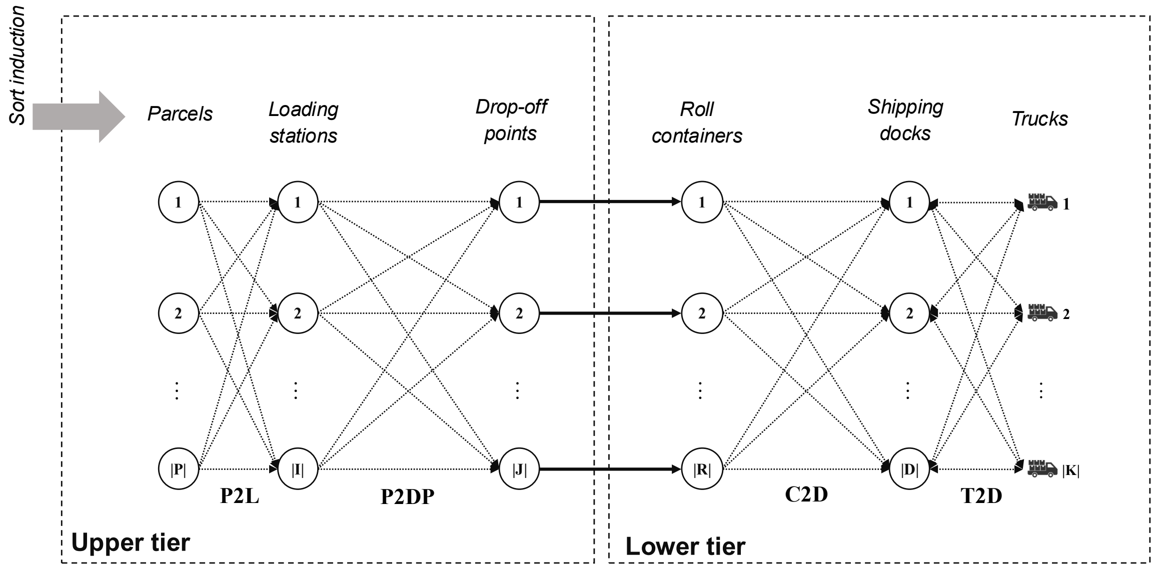


Figure 3: Graph representation of parcel flows within two-tier robotized sorting systems.

Note that the resulting sorting plan does *not* specify detailed multi-robot routing; these decisions are delegated to a lower-level, congestion-aware fleet control system that operates based on our assignment decisions. Our model instead focuses on determining static parcel routes and flows by minimizing travel within the facility, using the expected shortest travel times of robots moving between nodes in the sortation area. To mitigate congestion and deadlocks, we follow (Xu et al., 2022) and assume that aisles and cross-aisles are organized as two opposing unidirectional lanes.

Since layout design is beyond the scope of this study, we assume a predefined facility layout and robot path topology. Within this predefined layout, roll containers provide the physical interface between the upper and lower tiers via a spiral conveyor. In the model, each **roll container–drop-off-point pair** is represented as a **single node**, reflecting their **one-to-one correspondence**. As noted earlier, roll containers have limited capacity, which constrains the number of parcels they can hold; when demand exceeds this capacity, parcels bound for the same destination may be split across multiple roll containers, all of which are routed to the destination truck. The calculation of expected travel times for mobile robots (between loading stations and drop-off points) and for roll containers (to the shipping docks) is detailed in Appendix A.

4 Problem Modeling

In this section, we present a MIP model, referred to as the parcel sort planner (*Parcel*-SP), that holistically integrates all four assignment decisions described earlier. The goal of *Parcel*-SP is to generate parcel-level sorting plans that coordinate both upper- and lower-tier operations while minimizing total parcel travel time. Specifically, the objective function evaluates parcel travel time within the T-RSS, from induction to truck loading, serving as a proxy for throughput efficiency and energy-conscious robot utilization (Shen et al., 2023).

A path-based formulation is proposed for *Parcel*-SP to track the flow of each parcel throughout the facility. We leverage parcel-truck compatibility information $o_{p,k}$ to limit the paths considered for each parcel to only those that are feasible. Specifically, for each parcel p, we generate the set S_p of paths s that include only its compatible destination truck k.

We now present the parameters and decision variables used to formulate the *Parcel*-SP, summarized in Table 2.

Table 2: Notations for the *Parcel*-SP.

Sets	
I	Set of loading stations
J	Set of drop-off points (and corresponding roll containers)
D	Set of shipping docks
K	Set of trucks
P	Set of parcels
S	Set of all paths
S_p $(\subset S)$	Set of paths compatible with parcel p
Parameters	
$o_{p,k}$	1, if parcel p should be loaded onto truck k (parcel-truck compatibility); 0, otherwise
τ	Capacity of roll container (in parcels)
$\Delta^s_{j,k}$	1, if arc (roll container j, truck k) belongs to path s; 0, otherwise
$\Delta^s_{d,k}$	1, if arc (dock d, truck k) belongs to path s; 0, otherwise
t_i	Expected travel time from sort induction to loading station i by conveyor
$t_{i,j}$	Expected travel time from loading station i to drop-off point j by robot
$t_{j,d}$	Expected travel time from roll container j to shipping dock d by robot
κ_s	Transportation cost of a parcel along path s, assumed proportional to the transportation time $(t_i+t_{i,j}+t_{j,d})$.
Decision variables	
x^s_p	1, if parcel p is assigned to path s; 0, otherwise
$z_{j,d,k}$	1, if roll container j is assigned to dock d to fulfill truck k; 0, otherwise
$a_{d,k}$	1, if shipping dock d is assigned to truck k; 0, otherwise

The objective function (1) and constraints (2)–(9) together constitute the MIP formulation.

$$\text{Minimize} \quad \sum_{p\in P}\sum_{s\in S_p} \kappa_s x^s_p \tag{1}$$

$$\text{s.t.} \quad \sum_{s\in S_p} x^s_p = 1 \qquad \forall p\in P, \tag{2}$$

$$\sum_{d\in D}\sum_{k\in K} z_{j,d,k} \le 1 \qquad \forall j\in J, \tag{3}$$

$$z_{j,d,k} \le a_{d,k} \qquad \forall j\in J,\ \forall d\in D,\ \forall k\in K, \tag{4}$$

$$\sum_{p\in P}\sum_{s\in S_p} \Delta^s_{j,k} x^s_p \le \tau \sum_{d\in D} z_{j,d,k} \qquad \forall j\in J,\ \forall k\in K, \tag{5}$$

$$\sum_{p\in P}\sum_{s\in S_p} \Delta^s_{d,k} x^s_p \le \sum_{p\in P} o_{p,k} a_{d,k} \qquad \forall d\in D,\ \forall k\in K, \tag{6}$$

$$\sum_{d\in D} a_{d,k} = 1 \qquad \forall k\in K, \tag{7}$$

$$\sum_{k\in K} a_{d,k} \le 1 \qquad \forall d\in D, \tag{8}$$

$$x^s_p, z_{j,d,k}, a_{d,k} \in \{0,1\} \qquad \forall p\in P,\ \forall j\in J,\ \forall d\in D,\ \forall k\in K,\ \forall s\in S_p. \tag{9}$$

The objective function (1) minimizes the total parcel travel time as a proxy for minimizing sorting cost. Constraints (2) ensure that each incoming parcel p is assigned to a single path s from a loading station to a truck, enabling direct parcel movement to designated stations and subsequent transportation. Constraints (3) ensure that each roll container j is assigned parcels bound for only one truck destination; the inequality permits a container to remain unused if demand is insufficient. Constraints (4) couple decision variables to ensure the proper assignment of roll containers to shipping docks. Constraints (5) synchronize the upper and lower tiers by establishing a one-to-one correspondence between drop-off points and their assigned roll containers, subject to capacity constraints. Similarly, constraints (6) ensure that parcels take paths assigned to the designated truck and dock combination in the lower tier, meeting the demand of truck k. Constraints (7) and (8) regulate the relationship between shipping docks and trucks, ensuring each truck is assigned to one dock, and allow docks to remain idle if fewer trucks are available. Finally, constraints (9) specify the domains of the decision variables.

It is worth noting that, although the model is formulated at the parcel level, it considers a time-agnostic assignment of the entire parcel set, where the objective (total parcel travel time) does *not* depend on the processing sequence. If, instead, the objective were makespan or any other metric sensitive to the exact arrival or processing order, the model would need additional structure. In such settings, parcel arrival times must be explicitly modeled as inputs, and the formulation must include temporal constraints and sequencing dependencies so that scheduling decisions directly affect system performance.

To enhance solution efficiency, we incorporate Equation (10) to strengthen the model and accelerate convergence.

Proposition 1. *For Parcel-SP defined by* (1)–(9)*, the inequality*

$$\sum_{j\in J}\sum_{d\in D} z_{j,d,k} \geq \theta_k \qquad \forall k \in K, \tag{10}$$

is valid, where $\theta_k = \left\lceil \frac{\sum_{p\in P} o_{p,k}}{\tau} \right\rceil$ *is the minimum number of roll containers required to satisfy the demand for truck k.*

Proof. Constraint (5) in *Parcel*-SP limits each container to at most τ parcels destined for the same truck. Summing over all containers j assigned to truck k gives

$$\sum_{j\in J}\sum_{p\in P}\sum_{s\in S_p} \Delta^s_{j,k} x^s_p \leq \tau \sum_{j\in J}\sum_{d\in D} z_{j,d,k}.$$

By constraint (6) and the fact that each parcel p is assigned to exactly one path s, the left-hand side equals $\sum_{p\in P} o_{p,k}$, representing the total demand of parcels assigned to truck k. Dividing by τ and taking the ceiling to account for integer containers gives

$$\theta_k = \left\lceil \frac{\sum_{p\in P} o_{p,k}}{\tau} \right\rceil \leq \sum_{j\in J}\sum_{d\in D} z_{j,d,k},$$

establishing (10) as a valid lower bound on the total containers needed for truck k. □

Note that incorporating this bound into the optimization model reduces the search space and improves computational efficiency. This applies to formulations with different objective functions. For example, it holds as an equality when minimizing travel time or throughput time, as in our setting. In contrast, when minimizing makespan, it serves as a lower bound because additional roll containers may be used to balance resources (Jiang et al., 2021; Li et al., 2022).

This proposition also highlights the usefulness of using *aggregate* information. Drawing parallels to the concept of job families in Yang et al. (2022) and traditional bin packing lower bounds, this property shifts focus from *individual* parcels to *grouped* information, motivating the investigation of formulations defined at an *aggregated* level, coarser than the parcel level, as discussed next.

5 Solution Approach

While *Parcel*-SP integrates all relevant decisions to produce a detailed parcel-level sorting plan, solving it optimally is computationally prohibitive, especially for the large-scale instances encountered in practice. At the same time, operational environments require solution methods that can be rerun quickly as conditions evolve or disruptions occur.

Data availability further limits the use of *Parcel*-SP. Before the start of a sort shift, hub operators typically observe only aggregate workload information, such as the expected number of incoming parcels per truck and their destinations, whereas the full set of parcels P and their parcel-level attributes is revealed only after inbound trailers are unloaded and parcels are scanned. Collecting, transmitting, and optimizing over this parcel-level information in a time-sensitive environment can be burdensome, making *Parcel*-SP difficult to use as a fully anticipative planning tool.

Consequently, practitioners often simplify the decision process through rule-based schemes. For example, fixed allocation policies may assign parcels to loading stations as they arrive to the hub, and zoning strategies may *a priori* define relationships between loading stations and drop-off points (Zou et al., 2021). Such rules yield computationally- and data-light baselines in which remaining decisions are handled greedily or via local optimization. However, these approaches limit coordination, and their performance deteriorates with increasing parcel volume, as later shown in our experiments. To address these challenges, we propose a two-stage *Offline-then-Online* (OTO) framework that continues to model decisions in an integrated manner while remaining computationally tractable and aligned with operational decision timing and data availability.

5.1 Offline-then-Online framework

The approach begins with an *offline* stage that computes a commodity-based sorting plan before parcels arrive. A *commodity* is defined as a group of parcels associated with a truck k and destined for a specific delivery area. This offline stage uses the *actual aggregate demand*: for each incoming truck, the number of parcels that must go to each destination is known before the start of the sort wave, but not the individual parcel details. In the subsequent *online* stage, parcels are scanned upon arrival to identify their commodity type and are then assigned in real time to a path according to a simple control policy derived from the precomputed offline plan. Figure 4 illustrates the idea on a small example with 10 parcels and three commodities.

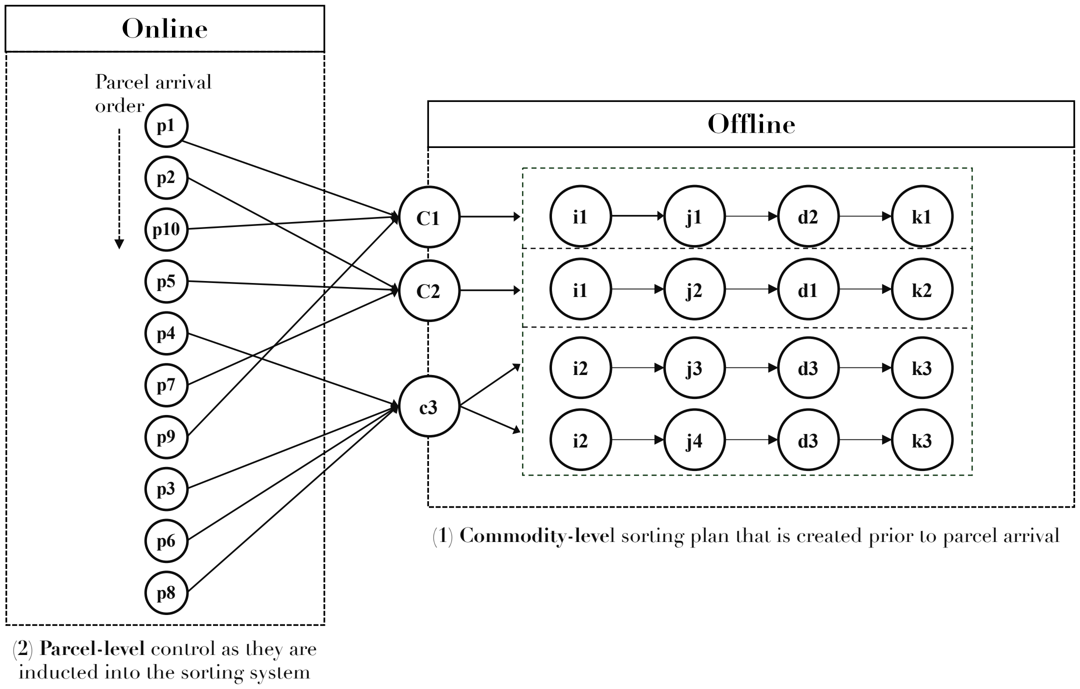

Online
Parcel arrival order
p1
p2
p10
p5
p4
p7
p9
p3
p6
p8
C1
C2
c3
Offline
i1
j1
d2
k1
i1
j2
d1
k2
i2
j3
d3
k3
i2
j4
d3
k3
(1) Commodity-level sorting plan that is created prior to parcel arrival
(2) Parcel-level control as they are inducted into the sorting system

Figure 4: Illustration of the proposed framework

In the remainder of this section, we discuss the proposed solution approach for both the offline planning stage, and the online control stage, which we assess later in the experiments section.

5.2 The offline phase

We develop two methods for the offline stage. The first is an exact approach that mirrors the structure of *Parcel*-SP but operates on aggregated (commodity-level) flows rather than individual parcels. This aggregation significantly reduces problem complexity, ensuring computational tractability for large-scale instances. The second is a heuristic method designed for large-scale layouts or environments without access to commercial solvers. It leverages problem characteristics at the commodity level to efficiently generate high-quality solutions.

5.2.1 Exact method

We propose an alternative MIP model, the commodity sort planner (*Commodity*-SP), which is less granular than *Parcel*-SP and shifts decisions from individual parcels to aggregate flows per truck (commodity). Specifically, we introduce w_k (total parcels for truck k) in place of parcel–truck indicators $o_{p,k}$, and replace parcel-level assignment variables x_p^s with path flows h_k^s for commodity k, alongside path-usage binaries x_k^s. Lastly, similar to *Parcel*-SP, we generate paths $s \in S_k$ for each commodity k, considering only paths that share the same parcel destination, i.e., the truck's delivery zone. The corresponding MIP formulation for *Commodity*-SP is described next.

$$\text{Minimize} \quad \sum_{k \in K} \sum_{s \in S_k} \kappa_s \, h_k^s \tag{11}$$

$$\text{s.t.} \quad h_k^s \leq \tau \, x_k^s \qquad \forall k \in K, \; \forall s \in S_k, \tag{12}$$

$$\sum_{k \in K} \sum_{s \in S_k} \Delta_{j,k}^s \, x_k^s \leq 1 \qquad \forall j \in J, \tag{13}$$

$$\sum_{s \in S_k} \Delta_{d,k}^s \, h_k^s \geq w_k \, a_{d,k} \qquad \forall d \in D, \; \forall k \in K, \tag{14}$$

$$\sum_{d \in D} a_{d,k} = 1 \qquad \forall k \in K, \tag{15}$$

$$\sum_{k \in K} a_{d,k} \leq 1 \qquad \forall d \in D, \tag{16}$$

$$x_k^s \leq h_k^s \qquad \forall k \in K, \; \forall s \in S_k, \tag{17}$$

$$x_k^s, \; a_{d,k} \in \{0,1\} \qquad \forall d \in D, \; \forall k \in K, \; \forall s \in S, \tag{18}$$

$$h_k^s \in \mathbb{R}_+ \qquad \forall k \in K, \; \forall s \in S. \tag{19}$$

Objective (11) minimizes the total volume-weighted travel time over chosen paths. Constraints (12) link the flow variables to path-usage binaries while enforcing the roll-container capacity τ. Constraint (13) limit each roll container j to at most one used path. Constraint (14) ensure that the demand of truck k is routed through its assigned dock, and (15)–(16) handle truck-to-dock assignments. Finally, Constraints (17) are included as valid inequalities to strengthen the formulation.

We now formalize the relationship between *Parcel*-SP and *Commodity*-SP. Conceptually, *Commodity*-SP *aggregates* parcel-level decisions into truck–path flows: instead of routing individual parcels, it specifies how many parcels from each truck are assigned to each path. This aggregation is obtained by summing parcel–path variables within each truck destination, following ideas typically used in multi-commodity network design (Crainic, 2000). However, to operationalize such a plan in practice, it must be disaggregated back to parcel-level assignments. While arbitrary fractional flows cannot generally be mapped to discrete parcels, classical flow-decomposition results (Ahuja et al., 1993) imply that integral, balanced flows can be decomposed into unit paths. We therefore establish conditions under which a solution to *Commodity*-SP admits an exact disaggregation into a parcel-feasible solution of *Parcel*-SP, and discuss a recovery procedure otherwise.

We first introduce the notation and assumptions used in our analysis. For each truck $k \in K$, define its associated (compatible) parcel set by $P_k := \{p \in P : o_{p,k} = 1\}$. For any parcel-level routing $x = (x_p^s)$, we define the aggregated flows $h = \mathcal{H}(x)$ by

$$h_k^s := \sum_{p \in P_k} x_p^s \quad \text{for all } k \in K, \; s \in S_k.$$

We make the following assumptions:

(A1) Each parcel belongs to a unique truck, so $(P_k)_{k\in K}$ forms a partition of P.

(A2) For each truck $k \in K$, all parcels $p \in P_k$ share the same set of admissible paths, i.e., $S_p = S_k$ for all $p \in P_k$.

(A3) Each truck load is known and equals the number of its compatible parcels, i.e., $w_k = |P_k|$ for all $k \in K$.

Theorem 1 (Parcel–commodity equivalence)**.** *Let $\mathcal{F}_{\mathrm{PSP}}$ and $\mathcal{F}_{\mathrm{CSP}}$ denote the feasible regions of* Parcel-*SP and* Commodity-*SP, respectively, and assume* (A1)–(A3) *hold, and that path costs are separable.*

(i) Aggregation (projection). *For any parcel-feasible solution $(x, z, a) \in \mathcal{F}_{\mathrm{PSP}}$, its aggregated flows $h := \mathcal{H}(x)$, together with the same dock assignment a, can be extended to a commodity-feasible solution $(h, \bar{x}, a) \in \mathcal{F}_{\mathrm{CSP}}$ for some $\bar{x}$.*

(ii) Cost preservation. *For any parcel-feasible solution $(x, z, a) \in \mathcal{F}_{\mathrm{PSP}}$ and $h := \mathcal{H}(x)$,*

$$\sum_{p\in P}\sum_{s\in S_p} \kappa_s x_p^s = \sum_{k\in K}\sum_{s\in S_k} \kappa_s h_k^s.$$

Moreover, Commodity-*SP provides a lower bound on* Parcel-*SP: $z^*_{\mathrm{CSP}} \le z^*_{\mathrm{PSP}}$.*

(iii) Disaggregation (lifting and exactness). *Let $(\bar{h}, \bar{x}, \bar{a}) \in \mathcal{F}_{\mathrm{CSP}}$ have integral, balanced, dock-consistent flows, i.e.,*

$$\bar{h}_k^s \in \mathbb{Z}_+, \qquad \sum_{s\in S_k} \bar{h}_k^s = w_k, \qquad \sum_{s\in S_k} \Delta_{d,k}^s \bar{h}_k^s = w_k \bar{a}_{d,k} \quad \forall k \in K,\ d \in D.$$

Then there exists a parcel-feasible solution $(\hat{x}, \hat{z}, \bar{a}) \in \mathcal{F}_{\mathrm{PSP}}$ whose aggregation matches $\bar{h}$, i.e., $\mathcal{H}(\hat{x}) = \bar{h}$, and whose objective value coincides with that of $(\bar{h}, \bar{x}, \bar{a}) \in \mathcal{F}_{\mathrm{CSP}}$. Moreover, such a disaggregation can be obtained in $O(|P|)$. Hence, if $(\bar{h}, \bar{x}, \bar{a})$ is optimal for Commodity-*SP, then $z^*_{\mathrm{CSP}} = z^*_{\mathrm{PSP}}$.*

Proof. (i) *Aggregation (projection).* Let $(x, z, a) \in \mathcal{F}_{\mathrm{PSP}}$ be arbitrary and define $h := \mathcal{H}(x)$ by

$$h_k^s = \sum_{p\in P_k} x_p^s, \qquad k \in K,\ s \in S_k.$$

Under (A1)–(A3), parcels can be grouped by their unique truck k and all parcels in P_k share the same path set S_k. The commodity formulation (*Commodity*-SP) is obtained by summing the parcel-level constraints of *Parcel*-SP over $p \in P_k$ and substituting $\sum_{p\in P_k} x_p^s$ by h_k^s. Therefore, for the given (x, z, a), the aggregated pair (h, a) satisfies all flow-balance, capacity, and dock-assignment constraints of *Commodity*-SP. By choosing a compatible path-usage vector $\bar{x}$ (e.g., setting $\bar{x}_k^s = 1$ whenever $h_k^s > 0$ and $\bar{x}_k^s = 0$ otherwise), we obtain a triple $(h, \bar{x}, a) \in \mathcal{F}_{\mathrm{CSP}}$, as claimed.

(ii) *Cost preservation.* By the definition of $\mathcal{H}(x)$ and the path-separable cost structure,

$$\sum_{p\in P}\sum_{s\in S_p} \kappa_s\, x_p^s = \sum_{k\in K}\sum_{p\in P_k}\sum_{s\in S_k} \kappa_s\, x_p^s = \sum_{k\in K}\sum_{s\in S_k} \kappa_s \Big(\sum_{p\in P_k} x_p^s\Big) = \sum_{k\in K}\sum_{s\in S_k} \kappa_s\, h_k^s.$$

Thus aggregation preserves the objective value of any given parcel-feasible solution. Together with part (i), this implies that *Commodity*-SP optimizes over a relaxation of the projection of $\mathcal{F}_{\mathrm{PSP}}$ without changing the objective on projected points, and hence $z^*_{\mathrm{CSP}} \leq z^*_{\mathrm{PSP}}$.

(iii) *Disaggregation (lifting and exactness).* Consider any feasible solution $(\bar{h}, \bar{x}, \bar{a}) \in \mathcal{F}_{\mathrm{CSP}}$ satisfying the given integrality, balance, and dock-consistency conditions. By (A1) and (A3), each truck k has an associated parcel set P_k with $|P_k| = w_k$, and the identity $\sum_{s \in S_k} \bar{h}^s_k = w_k$ implies that the nonnegative integers $\{\bar{h}^s_k : s \in S_k\}$ form a partition of the parcels of truck k.

A disaggregated parcel assignment $\hat{x}$ can be constructed as follows. Fix $k \in K$ and enumerate the parcels in P_k arbitrarily. Then iterate over the paths $s \in S_k$ and assign the next $\bar{h}^s_k$ parcels in this list to path s, setting $\hat{x}^s_p = 1$ for these pairs and $\hat{x}^s_p = 0$ otherwise. This guarantees that exactly $\bar{h}^s_k$ parcels of truck k use path s and that each parcel is assigned to exactly one path, so $\sum_{p \in P_k} \hat{x}^s_p = \bar{h}^s_k$ and hence $\mathcal{H}(\hat{x}) = \bar{h}$. The work required for truck k is $O(|P_k|)$, and summing over all trucks gives a total runtime $O\big(\sum_{k \in K} |P_k|\big) = O(|P|)$. Equivalently, one may view this construction as solving, for each truck k, the transportation problem described in Appendix B. Since the problems are separable in k and $\sum_{k \in K} |P_k| = |P|$, the total computational effort is $\sum_{k \in K} O(|P_k|) = O(|P|)$.

Furthermore, since the capacity and dock constraints (5)–(6) depend on parcel assignments solely through the aggregate flow $\mathcal{H}(x)$, the feasibility of $\bar{h}$ in *Commodity*-SP guarantees the feasibility of $\hat{x}$ in *Parcel*-SP. By setting $\hat{z}$ consistent with $\bar{x}$ (i.e., activating roll containers for used paths), we obtain a feasible solution $(\hat{x}, \hat{z}, \bar{a}) \in \mathcal{F}_{\mathrm{PSP}}$. By part (ii), the objective values match. Thus, $z^*_{\mathrm{CSP}} = z^*_{\mathrm{PSP}}$.

□

Remark (Recovery under deviations). When the integrality and balance conditions in Theorem 1(iii) do not hold, for example, due to fractional flows from a relaxation of *Commodity*–SP, incomplete path enumeration, or mismatches between planned truck demand and realized parcel inflow, the lifting result no longer guarantees a feasible parcel-level solution. In such cases, the assignment model in Appendix B can be augmented with a high-penalty dummy path to flag overflow parcels that cannot be routed on the precomputed paths. The resulting overflow may then be handled by re-optimizing a smaller *Commodity*–SP on the *residual* demand or by a simple greedy or heuristic policy; a recovery procedure is described in Appendix B.

Implementation note. In our computational experiments, these deviations did not arise: the demands w_k and capacity τ were integer-valued, the path-generation procedure ensured sufficient capacity for each truck, and *Commodity*–SP was solved as a MIP with integral path flows h^s_k. Thus the conditions of Theorem 1(iii) were satisfied, and every offline commodity plan could be mapped directly to a feasible parcel-level solution via the assignment procedure in Appendix B, without invoking the recovery mechanism. We view these assumptions as mild and broadly consistent with typical parcel-sorting operations (e.g., integer workload counts and capacity planning that targets feasible truck loads). In the deployed implementation, the assignment model is used only for infeasibility detection and diagnostic purposes if needed; real-time routing decisions are made by the online procedure described later in Section 5.3, which is simple to implement and better suited to streaming parcel arrivals.

5.2.2 Heuristic method

Although the *Commodity*-SP is more efficient than the *Parcel*-SP, solving large-scale instances with standard solvers remains computationally demanding. To address this challenge, we propose a constructive heuristic that builds on Proposition 1 and Theorem 1. This approach offers a more intuitive, problem-specific alternative to common bio-inspired methods, such as genetic algorithms or particle swarm optimization, while avoiding the burden of extensive parameter tuning.

The proposed heuristic aims to reduce total parcel travel time by prioritizing high-volume commodities on shorter paths, while also minimizing the number of roll containers to improve resource utilization. It respects operational constraints—including roll container capacity, dock availability, and drop-off point usage—ensuring that all assignments remain feasible within the T-RSS.

The algorithm takes as input the set of commodity destinations, their actual aggregate demand, operational capacities, and a set of preprocessed l-shortest paths. It operates in two phases. In the first phase, commodities with the largest parcel volumes are prioritized. Each commodity is assigned the minimum number of full roll containers required, and these are allocated to the shortest feasible paths, with ties broken randomly. Only full containers are allocated at this stage, while any remaining parcels are deferred to the second phase. In the second phase, commodities with residual parcels are considered in descending order of remaining volume and are assigned to the shortest feasible paths, again with random tie-breaking when multiple options exist.

To improve solution quality, the heuristic employs a multi-start procedure with randomized path selection. In the initial run, trucks are ordered by descending demand, and each container is assigned to the shortest available path, producing a greedy baseline solution. In subsequent runs, stochasticity is introduced: truck orderings are randomized, and path selection allows the second-shortest path to be chosen with 50% probability ($l = 2$). Across multiple runs, the best solution is retained, defined as the assignment yielding the lowest total parcel travel time.

This strategy combines exploitation of efficient paths with exploration of near-optimal alternatives, thereby reducing the risk of premature convergence to local minima. At the same time, it remains computationally lightweight, straightforward to implement, and intuitive for practitioners, making it well suited for warehouse planning and decision support. The full procedure is summarized in Algorithm 1.

5.3 The Online phase

The offline plan determines a set of *candidate* paths for each commodity, where each path specifies a sequence of nodes—loading station, drop-off point, roll container, shipping dock, and truck—through which the associated

Algorithm 1 Heuristic offline planning

```
1: final_assignments, used_docks, total_travel_time $\leftarrow \emptyset$; best_solution $\leftarrow (\emptyset, \infty)$; initial_run $\leftarrow$ True
2: for run $= 1$ to num_runs do
3:     truck_order $\leftarrow \begin{cases} \text{sort}(\{k\}, \downarrow \theta_k;\ \text{tie: demand}) & \texttt{initial\_run} \\ \text{randperm}(\{k\}) & \text{otherwise} \end{cases}$
4:     initial_run $\leftarrow$ False; current_solution $\leftarrow \emptyset$; current_cost $\leftarrow 0$
5:     Requirements:
6:     for $k \in$ truck_order do
7:         compute full$_k$ (required full containers), resid$_k$ (residual volume)
8:     end for
9:     Assign full containers:
10:    while $\sum_k$ unassigned_full$_k > 0$ do
11:        for $k \in$ truck_order do
12:            if unassigned_full$_k = 0$ then continue
13:            end if
14:            $\mathcal{S}_k \leftarrow$ {feasible paths for $k$}
15:            $s_{(1)} \leftarrow \arg\min_{s \in \mathcal{S}_k} \kappa_s$;  $s_{(2)} \leftarrow$ 2nd-min$_{s \in \mathcal{S}_k} \kappa_s$
16:            $r \sim \mathcal{U}(0,1)$;  $s^\star \leftarrow \begin{cases} s_{(1)}, & r < 0.5 \\ s_{(2)}, & \text{otherwise} \end{cases}$
17:            assign one full container of $k$ to $s^\star$; update final_assignments, used_docks, total_travel_time
18:        end for
19:    end while
20:    Assign residuals:
21:    for $k$ in sort($\{k\}, \downarrow$ resid$_k$) do
22:        $\mathcal{S}_k \leftarrow$ {feasible paths for $k$};  $s_{(1)}, s_{(2)}$ as above;  $r \sim \mathcal{U}(0,1)$;  $s^\star \leftarrow \begin{cases} s_{(1)}, & r < 0.5 \\ s_{(2)}, & \text{otherwise} \end{cases}$
23:        assign residual container/volume of $k$ to $s^\star$; update final_assignments, total_travel_time
24:    end for
25:    current_cost $\leftarrow$ cost(total_travel_time)
26:    if current_cost $<$ best_solution.cost then
27:        best_solution $\leftarrow$ (current_solution, current_cost)
28:    end if
29: end for
30: return best_solution
```

parcel volume is routed. Once the offline plan is established, an online control algorithm (*Online*-SP) is executed to dynamically manage parcels as they enter the sorting system.

The detailed procedure of *Online*-SP is given in Algorithm 2. As parcels arrive, the transportation cost of each candidate path for parcel p is computed as the sum of the expected travel times from sort induction to the loading station (t_i), from the loading station to the drop-off point or roll container ($t_{i,j}$), and from the roll container to the shipping dock ($t_{j,d}$). If multiple parcels destined for the same truck arrive consecutively, they are assigned iteratively, beginning with the longest candidate paths and progressing to shorter ones while avoiding duplicate assignments. This descending longest-processing-time rule is desirable in this setting, as it helps mitigate drop-off point occupancy and congestion (Chen et al., 2024b). It is worth noting, however, that this algorithm can be adapted to different operational objectives by incorporating other dispatching rules, such as shortest-processing-time first, and can accommodate additional constraints as needed.

Algorithm 2 *Online*-SP: Parcel-level sort control with offline quotas

```
1: $\texttt{Assignments} \leftarrow \emptyset$
2: for all $k \in K$ and $s \in S_k^*$ do
3:     $r_k^s \leftarrow h_k^s$
4: end for
5: for each arriving parcel $p$ do
6:     $k \leftarrow \text{commodity}(p)$
7:     $\texttt{Candidates}_p \leftarrow \{\, s \in S_k^* \mid r_k^s > 0 \ \wedge \ \text{cap}(s) \geq 1 \,\}$
8:     if $\texttt{Candidates}_p \neq \emptyset$ then
9:         $t_s \leftarrow t_i + t_{i,j} + t_{j,d}$ for all $s \in \texttt{Candidates}_p$
10:        $s^\star \leftarrow \arg\max_{s \in \texttt{Candidates}_p} t_s$
11:        assign $p \to s^\star$;   $r_k^{s^\star} \leftarrow r_k^{s^\star} - 1$;   update $\text{cap}(s^\star)$
12:        $\texttt{Assignments} \leftarrow \texttt{Assignments} \cup \{(p, s^\star)\}$
13:    end if
14: end for
15: return $\texttt{Assignments}$
```

6 Numerical Experiments

This section examines the computational performance of our optimization model (*Parcel*-SP) and solution framework (OTO approach) and investigates the value of decision integration. Section 6.1 describes test instance generation. Section 6.2 compares *Parcel*-SP, OTO approach, and rule-based assignments in terms of solution quality and computational time. Section 6.3 analyzes how different levels of decision integration within T-RSS affect overall performance. All analyses and experiments were conducted using Python (v3.9.6) on an M1 processor laptop with 16 GB of RAM, with Gurobi (v10.0) as the commercial optimization solver.

6.1 Description of test instances

Delivery station sizes and parcel volumes vary across regions (Rodrigue, 2020). To capture this variability, we design computational experiments using small, medium, and large-sized layouts. Since the primary focus of this study is not on layout design, we assume a basic layout derived from previous studies (Chen et al., 2024b; Xu et al., 2022). Appendix A provides a detailed discussion of how the assumed layout influences expected moving distances and robot travel times. Furthermore, the proposed solution procedures are adaptable to a variety of layouts, provided that the corresponding T-RSS distance matrices are specified.

The large-sized layout reflects realistic industrial cases from Xu et al. (2022), while the small and medium layouts are proportionally scaled down. Whereas prior studies have primarily focused on upper-tier layouts to validate solutions, we extend our approach by incorporating lower-tier details. In our setting, the lower tier models practical delivery station operations, where roughly a dozen trucks arrive during each sort wave to load parcels that have been pre-sorted by destination for delivery (Dive, 2020).

Without loss of generality, we assume that the loading stations are arranged in a single row at regular intervals along the north side of the system, adjacent to a conveyor belt that transports parcels from the entrance to the loading area. At the induction point, a scanner determines the appropriate loading station for each parcel based on the generated assignment.

Since P2L assignment is critical to our optimization, the travel time on the conveyor belt is also included (Chen et al., 2024b; Xu et al., 2022). Although loading speeds can vary, we assume that fast robotic arms ensure efficient parcel loading within the inter-arrival times of incoming parcels (Amazon, 2025b). The inter-arrival times are assumed to be at least 3 seconds to reduce the risk of service capacity bottlenecks (Boysen et al., 2023).

At the demand side, we associate each parcel with a commodity that corresponds to its designated delivery truck, as determined by the parcel–truck compatibility parameter $o_{p,k}$. For each truck k, parcel demand is modeled as an integer-valued normal random variable with a mean μ uniformly drawn between 100 and 300, and a standard deviation σ between 0 and 50. This choice reflects typical delivery truck capacities (200–300 parcels per truck; Figliozzi (2017)) while ensuring realistic variability and integer-valued demands. The roll container capacity τ is fixed at 40 parcels to mirror practical handling constraints. A summary of the key parameter settings used in generating the test instances is provided in Table 3.

Table 3: Layout sizes and corresponding parameter settings.

Layout size	**Loading stations**	**Drop-off points**	**Shipping docks**	**Commodities (trucks)**
	$\lvert\mathbf{I}\rvert$	$\lvert\mathbf{J}\rvert$	$\lvert\mathbf{D}\rvert$	$\lvert\mathbf{K}\rvert$
Small	2	24 (4×6)	3	$U[2, 3]$
Medium	4	48 (4×12)	6	$U[4, 6]$
Large	6	108 (6×18)	14	$U[11, 14]$

6.2 Computational performance

To facilitate a comprehensive performance comparison across different layout sizes, we generate 15 test instances with randomized parcel volumes and sequences, resulting in a total of 45 unique test cases. For each instance, the number of trucks and their associated demands are randomly determined following Table 3 to define parcel–truck compatibility. A runtime limit of 3,600 seconds is imposed to reflect the short decision-making cycles typical of compact and flexible facilities. The following approaches are then compared:

- ***Parcel*-SP:** This approach uses model (1)–(10) to generate parcel-level plans.
- **OTO (Exact):** This approach implements the OTO framework described earlier, solving the offline stage using *Commodity*-SP (modeled in (11)–(19)), followed by *Online*-SP (Algorithm 2) for real-time control.
- **OTO (Heu):** Identical to OTO (Exact), but replaces the offline *Commodity*-SP optimization with the heuristic described in Algorithm 1. The hyperparameter for OTO (Heu), namely the number of runs, is set to 50, 100, and 200 for small, medium, and large instances, respectively.
- **RULE:** A sequential rule-based assignment procedure commonly adopted in real-world sorting operations, combining random and proximity-based selection mechanisms with minor adaptations from prior studies (Boysen et al., 2023; Chen et al., 2024b; Xu et al., 2022). Specifically, four decision stages are defined:

(1) *P2L*—incoming parcels are sequentially assigned to loading stations in a round-robin manner, ensuring balanced utilization; (2) *T2D*—trucks are randomly assigned to available docks with equal probability; (3) *P2DP*—given the above assignments, parcels are greedily allocated to the nearest feasible drop-off point or roll container, prioritizing already activated ones associated with the same truck while respecting container capacity τ; and (4) *C2D*—each activated roll container inherits the dock assigned to its corresponding truck. These combined rules offer high computational efficiency and ease of implementation without requiring optimization modeling. The detailed procedure is summarized in Appendix C.

The computational performance and results of these approaches are summarized in Table 4, reporting averages by layout size. The table presents: (1) Obj—average total travel time (in seconds); (2) CPU—average runtime (in seconds) capped at 3600 seconds; and (3) Gap—the percentage deviation from the best-found solution across all approaches. Detailed results for all test instances are provided in Appendix D.

Table 4: Comparison of computational performance across all solution approaches.

Layout size	***Parcel*-SP**		**OTO (Exact)**			**OTO (Heu)**			**RULE**		
	Obj(s)	CPU(s)	Obj(s)	Gap(%)	CPU(s)	Obj(s)	Gap(%)	CPU(s)	Obj(s)	Gap(%)	CPU(s)
Small	8,615.23	11.15	8,615.23	0.00	0.16	8,658.48	0.54	0.14	10,050.39	16.76	0.01
Medium	17,640.90	164.56	17,640.90	0.00	1.50	17,757.69	0.65	2.38	24,255.00	38.12	0.02
Large	-	3,600.00	54,535.91	0.00	3,600.00	55,336.75	1.46	80.98	82,672.89	52.25	0.08

Note: Obj indicates the average total parcel travel time as the performance metric, while CPU represents the average computational time required to generate the sorting plan. Gap refers to the average percentage difference between the best solution found. Note that for large-sized layouts, *Parcel*-SP fails to produce any feasible solution within the time limit.

The results in Table 4 provide the following key insights:

1. *Parcel*-SP integrates all decisions into a single optimization framework, assuming complete demand information at the individual parcel level. Although such information is rarely available in practice, the model serves as a benchmark for evaluating solution quality. This benchmark provides warehouse managers, whether considering the adoption of T-RSS or operating it without advanced operational optimization, with a reference metric. It illustrates the potential value that could be realized if the system were operated under ideal conditions, thereby offering practical guidance for investment decisions. From a computational perspective, *Parcel*-SP can generate optimal plans for small- to medium-sized layouts, but for large layouts it typically exceeds a one-hour runtime without producing any feasible solution.

2. OTO (Exact) computes valid lower bounds via the aggregated *Commodity*-SP and recovers feasible parcel-level solutions; in all tested instances these bounds coincided with the parcel optimum, and both were obtained at much shorter runtimes. For small- and medium-sized layouts, when combined with the *Online*-SP, it yields the same objective values as the *Parcel*-SP, and for large layouts it achieves near-optimal performance. Unlike

Parcel-SP, it can solve large layouts within the runtime limit. Nevertheless, its practicality may still be limited in certain facility settings. In such cases, OTO (Heu) provides a faster and more flexible alternative, particularly under stricter runtime constraints or when commercial solvers are unavailable.

3. While OTO (Heu) involves some performance trade-off, it delivers robust results across different instances, with average optimality gaps of 0.54% and 0.65% for small- and medium-sized layouts, respectively. For large layouts, the gap rises to 1.46% relative to OTO (Exact); however, OTO (Heu) is roughly 44 times faster, with an average runtime of only 80.98 seconds. Overall, the OTO framework—whether implemented with exact or heuristic methods—effectively minimizes total parcel travel time, even with less granular information.
4. Although RULE is computationally efficient, achieving runtimes under 0.1 seconds even for large-scale layouts, its performance deteriorates significantly as the problem size increases. The average percentage gap from the best-found solution is around 17%, 38%, and 52% for small, medium, and large layouts, respectively. These results suggest that RULE's independent, sequential decision-making lacks the coordination needed for complex sorting systems with interdependent decisions. This highlights the importance of integrating decisions across upper and lower tiers to optimize sorting processes simultaneously. Understanding such integration is therefore key to identifying improvement levers for intra-logistics and is discussed further in the next subsection.

6.3 The value of decision integration

This study extends previous research by not only validating findings on upper-tier decisions but also examining lower-tier decision dynamics in T-RSS and their effect on overall system performance. Lower-tier decisions are often omitted in prior work, based on the assumption that their consolidated flows (i.e., roll containers) exert less influence than the individual parcel processing in the upper tier. While valid to some extent, this perspective overlooks the system-wide benefits of synchronization. Integrating lower-tier decisions may offer additional insights, particularly in dense urban e-commerce settings where even minute-level improvements directly affect service quality.

This subsection therefore addresses this challenge by analyzing outcomes across different levels of decision integration. We hypothesize that simultaneous, integrated optimization of all decisions can improve the performance of T-RSS. By contrast, sequential approaches–where some decisions are fixed before optimizing the remaining ones–can compromise sorting efficiency, as interdependent decisions are not coordinated. To evaluate the impact of integration, we designed seven cases representing different levels of integration–optimization coupling, ranging from fully integrated scenarios, where all assignment decisions are optimized jointly, to cases in which decisions are made independently. The following summarizes how these JOINT variants–representing decisions optimized jointly across all tiers–were generated. Figure 5 presents these cases, omitting the familiar JOINT (1, 2, 3, 4) variants, for easier illustration.

- **JOINT (1, 2, 3, 4):** Decisions (1)–(4) are optimized simultaneously using *Parcel*-SP; for large instances, OTO (Exact) is applied.

- **JOINT (1, 2, 3):** Decision (4) is determined independently via T2D part of RULE and treated as a fixed input to *Parcel*-SP, which then optimizes decisions (1), (2), and (3) simultaneously.
- **JOINT (2, 3, 4):** Decision (1) is determined independently using P2L part of RULE, while decisions (2), (3), and (4) are optimized simultaneously through *Parcel*-SP.
- **JOINT(1, 2):** Decisions (3) and (4) are determined following the lower-tier part of RULE. *Parcel*-SP then optimizes decisions (1) and (2) with fixed decisions (3) and (4).
- **JOINT(2, 3):** Decisions (1) and (4) are determined independently using P2L and T2D parts of RULE, followed by the simultaneous optimization of decisions (2) and (3) through *Parcel*-SP.
- **JOINT (3, 4):** Decisions (1) and (2) are determined following the upper-tier part of RULE. *Parcel*-SP then optimizes decisions (3) and (4) with fixed decisions (1) and (2).
- **JOINT (0):** All decisions are made independently using RULE, without integration or optimization.

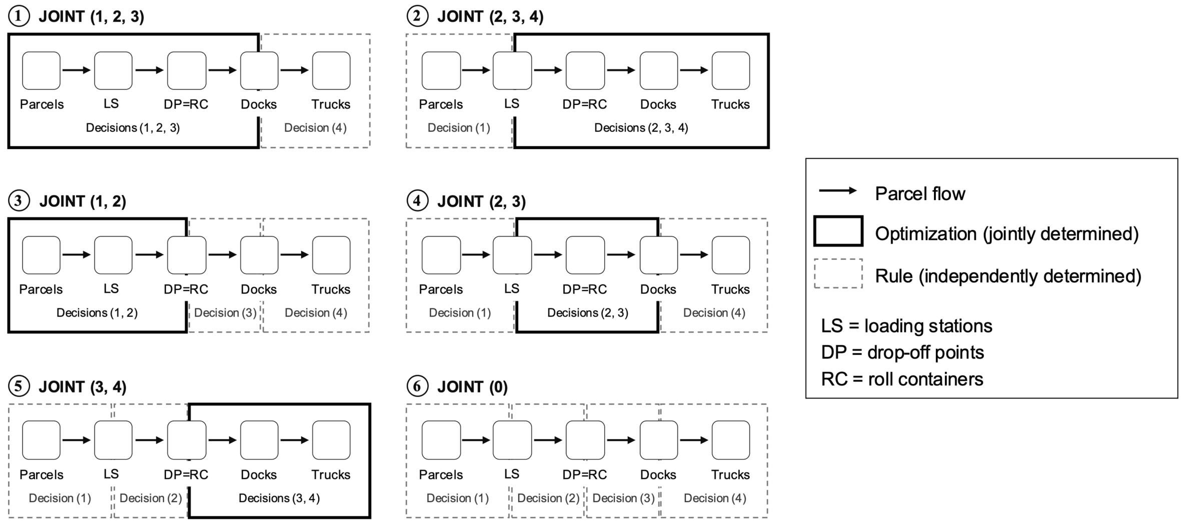


Figure 5: An illustration of different levels of integration. Note that Rule-based assignments are performed first, and after those decisions are fixed, optimization is carried out sequentially.

Solving decisions independently and fixing them in advance simplifies the problem and reduces computation time; however, our analysis focuses solely on evaluating performance trade-offs rather than computational efficiency. As demonstrated in the previous subsection, the proposed solution framework is already computationally efficient in generating effective sorting plans. Therefore, this analysis emphasizes the performance compromises caused by a lack of coordination among interdependent decisions, measured by total parcel travel time. We use the same test instances from the previous subsection, and all values reported below represent averages from 15 test instances for each layout size. Detailed results, including computational times for all instances, are provided in Appendix E.

6.3.1 Value of *full* integration

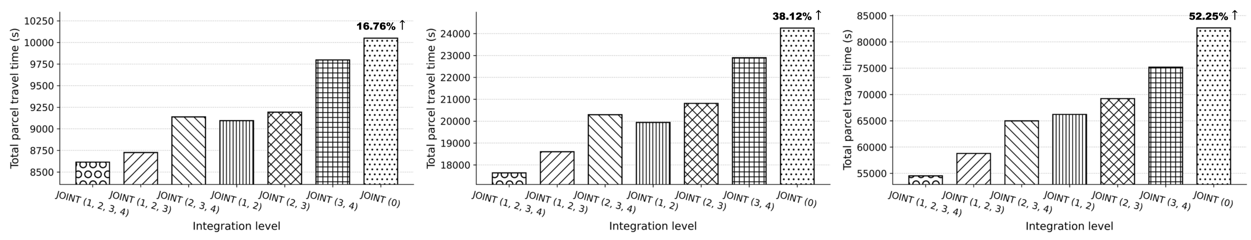
The overall results are summarized in Table 5, which explains the decisions included in each integration case and the resulting total parcel travel times across different layout sizes.

Table 5: Performance comparison across different integration cases.

Integration levels	**Decisions optimized simultaneously**				**Total parcel travel time (seconds)**		
	P2L	P2DP	C2D	T2D	**Small**	**Medium**	**Large**
JOINT (1, 2, 3, 4)	✓	✓	✓	✓	8,615	17,641	54,536
JOINT (1, 2, 3)	✓	✓	✓	✗	8,726	18,608	58,804
JOINT (2, 3, 4)	✗	✓	✓	✓	9,138	20,298	64,990
JOINT (1, 2)	✓	✓	✗	✗	9,095	19,942	66,217
JOINT (2, 3)	✗	✓	✓	✗	9,194	20,816	69,227
JOINT (3, 4)	✗	✗	✓	✓	9,798	22,901	75,211
JOINT (0)	✗	✗	✗	✗	10,050	24,255	82,673

Note: ✓ indicates the decision is integrated with at least one other decision in the optimization model, while ✗ indicates it is independently determined based on a predefined rule.

Across all tests, integrating decisions in any form consistently outperforms scenarios without integration. The performance gains become more pronounced as system size increases, since larger parcel volumes and resource requirements intensify system complexity and necessitate improved synchronization. Figure 6 further illustrates the pattern of increasing travel time as fewer decisions are jointly optimized. We highlight the percentage gap of JOINT (0) relative to JOINT (1, 2, 3, 4), where higher percentages indicate worse performance. For example, JOINT (0) yields parcel travel times that are approximately 17%, 38%, and 52% longer than those of JOINT (1, 2, 3, 4) in small, medium, and large layouts, respectively. Put differently, fully integrating decisions within the optimization model yields 14%, 27%, and 34% reductions in travel time across the three layout sizes—specifically, from 10,050 to 8,615 seconds, from 24,255 to 17,641 seconds, and from 82,673 to 54,536 seconds, respectively—compared to solving the same problem without joint optimization.



(A) Average results for the small-sized layout (B) Average results for the medium-sized layout (C) Average results for the large-sized layout

Figure 6: Performance comparison across integration levels and layout sizes.

6.3.2 Value of *partial* integration

Three-level integration For cases integrating *three* decisions, the results show varying yet substantial improvements over JOINT (0), with larger percentage improvement indicating better sorting performance. These gains are illustrated in Figure 7. In particular, JOINT (1, 2, 3)–which integrates all decisions except downstream T2D assignment–consistently outperforms JOINT (2, 3, 4), where upstream P2L assignment is excluded.

Since decisions (2) and (3) concern internal sorting operations, integrating the upstream decision together with these operations reduces parcel travel times by approximately 13%, 23%, and 29% for small, medium, and large layouts, respectively, whereas integrating the downstream decision with sorting yields reductions of only 9%, 16%, and 22% for the same layouts. This indicates that decision (1) (P2L assignment) is more critical to sorting performance than decision (4) (T2D assignment), consistent with Xu et al. (2022), who highlight the pivotal role of P2L assignments in T-RSS.

Figure 8 provides a direct comparison of the marginal performance impact obtained by excluding decisions (1) and (4), respectively, from JOINT (1, 2, 3, 4). P2L assignment governs how parcels enter the system, shaping overall flow and influencing all subsequent decisions. By contrast, T2D assignment offers limited leverage in compact layouts, where the small number of docks and trucks constrains potential gains. Its role, however, becomes more significant in larger facilities, where higher parcel volumes require additional trucks and more frequent delivery waves. While often overlooked, T2D assignment should not be disregarded in urban last-mile delivery settings, as repeated waves accumulate downstream effects.

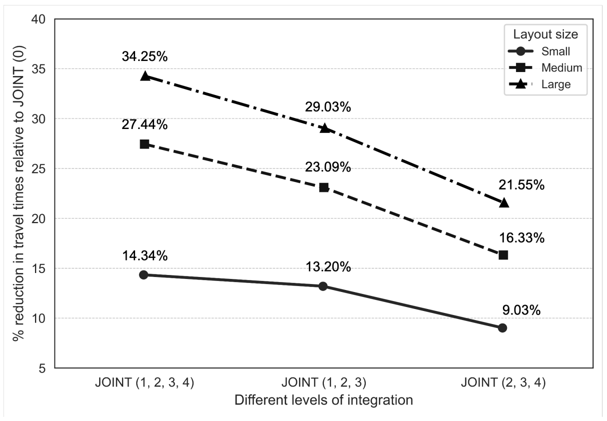


Figure 7: Performance benefits of optimizing three or more integrated decisions compared to JOINT (0).

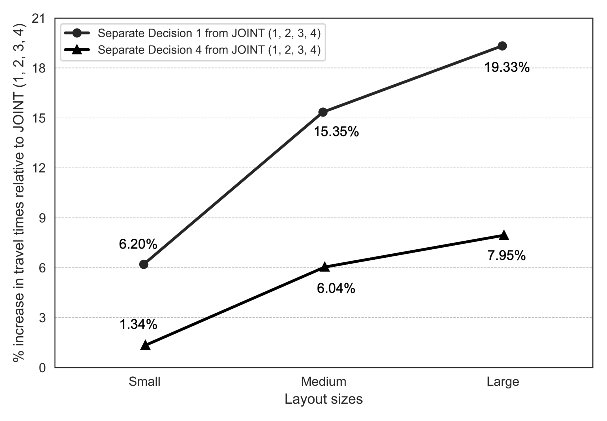


Figure 8: Impact of separating decisions 1 and 4 from JOINT (1, 2, 3, 4) on performance.

Two-level integration Figure 9 compares the performance of cases where two decisions are optimized jointly against the non-integrated baseline; larger percentages correspond to greater reductions in total travel time. Among these, JOINT (1, 2) delivers the largest improvement, followed by JOINT (2, 3) and JOINT (3, 4). This pattern indicates that upper-tier decisions have the strongest impact on performance, sortation-floor decisions come next, and lower-tier decisions have the smallest marginal effect. This aligns with prior studies (Chen et al., 2024b; Xu et al., 2022; Zhao

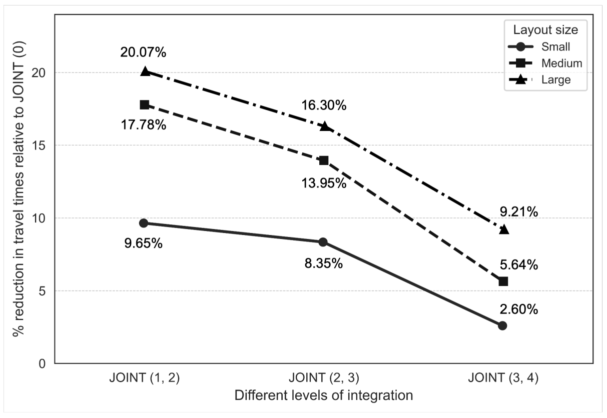


Figure 9: Performance benefits of optimizing two integrated decisions compared to JOINT (0).

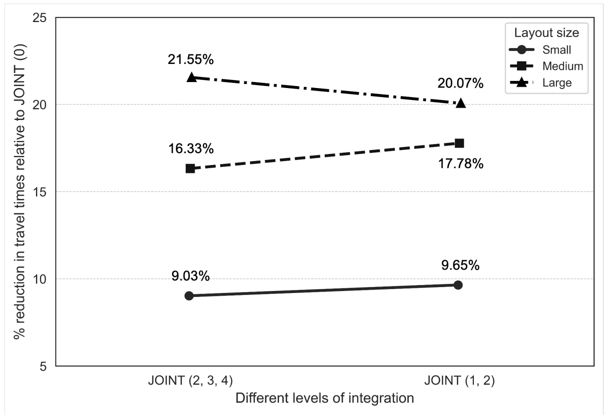


Figure 10: Performance benefits of JOINT (2, 3, 4) and JOINT (1, 2) compared to JOINT (0).

et al., 2024) which have emphasized the importance of decisions (1) and (2) as upstream decisions that interact directly with inbound operations.

The comparison between JOINT (1, 2) and JOINT (2, 3) is particularly revealing. JOINT (2, 3) optimizes only internal sortation-floor operations, yet it yields smaller gains than JOINT (1, 2), which includes P2L assignment. Although sortation-floor control may appear to be the most natural focus for robotic optimization, these results show that neglecting how parcels are launched into the system overlooks substantial benefits: effective initial P2L assignment is a key driver of overall performance.

Lower-tier integration While upper-tier decisions exert the greatest overall influence, the results also show that lower-tier optimization becomes increasingly important in large layouts and should not be overlooked in practice. Two implications follow. First, for facilities currently operating under JOINT (0), moving directly to a fully integrated upper-tier solution can be difficult, as it requires managing all parcel-level controls simultaneously. In such cases, practitioners may still unlock substantial improvements by focusing on relatively simpler lower-tier adjustments. For example, parcel travel times can be reduced by up to 9.21% as shown in Figure 9. Second, the comparison between JOINT (2, 3, 4) (partial synchronization of upper- and lower-tier decisions) and JOINT (1, 2) (upper-tier integration only) in Figure 10 illustrates the additional value of incorporating the lower tier once some upper-tier coordination is in place. In small- and medium-sized layouts, JOINT (1, 2) slightly outperforms JOINT (2, 3, 4), despite involving fewer integrated decisions. In larger systems, however, this relationship reverses: JOINT (2, 3, 4) reduces parcel travel times by roughly 22% relative to JOINT (0), compared with about 20% for JOINT (1, 2). This shift appears to stem from stronger interactions between upper- and lower-tier decisions as additional sorting resources (e.g., drop-off points and roll containers) are introduced. Overall, the results suggest that in multi-tier facilities—not limited to T-RSS—jointly coordinating tier operations can yield meaningful performance gains compared with optimizing each tier in isolation.

To summarize, the best performance is achieved with JOINT (1, 2, 3, 4), which fully integrates all four decisions. If only part of the system can be optimized in practice, prioritizing upper-tier decisions is usually the most effective starting point, mainly because P2L assignments strongly shape how parcels flow through the system. That said, the results also show that the real performance gains come from treating the system end-to-end: aligning how parcels enter (P2L), how they are routed and consolidated internally (P2DP and C2D), and how they are finally loaded (T2D). In practical terms, robotized systems deliver their full potential only when these stages are coordinated jointly, rather than optimized in isolation.

7 Concluding Remarks

This study examines four key assignment decisions within T-RSS for parcel delivery: (1) parcel-to-loading station, (2) parcel-to-drop-off point, (3) roll container-to-shipping dock, and (4) truck-to-shipping dock. Unlike prior research that focuses primarily on upper-tier operations, this study is the first to optimize the end-to-end process by synchronizing upstream parcel allocation with downstream outbound truck–dock assignment. We develop *Parcel*-SP as a theoretical benchmark based on detailed demand information. To improve computational tractability, we propose an offline–then–online (OTO) framework that decouples decision making and better reflects the information structure available in practice. In the offline stage, a deterministic, commodity-based sorting plan is computed using either an exact or a heuristic method; in the online stage, parcels are routed through the hub based on the precomputed candidate paths.

Numerical experiments across multiple layout sizes demonstrate the effectiveness of the proposed solution approaches. Parcel-level modeling yields benchmark-optimal travel times but becomes computationally intractable for large layouts, whereas more aggregate, commodity-based modeling offers a better trade-off between solution quality and computational effort. Consistent with this observation, *Parcel*-SP delivers benchmark-optimal solutions, while the proposed offline–then–online (OTO) framework achieves greater scalability. Within OTO, the exact variant attains optimal solutions within reasonable runtimes using the aggregate *Commodity*-SP formulation, whereas the heuristic variant scales more effectively with only slightly larger optimality gaps. Accordingly, OTO (Exact) is suitable when commercial solvers with moderate runtime budgets are available, while OTO (Heu) provides a practical alternative for larger-scale instances or settings without access to commercial solvers. In contrast, rule-based methods, though computationally efficient, exhibit substantial performance gaps, highlighting the need for optimization-based approaches. To assess decision integration, we compare fully and partially integrated schemes and find that upper-tier decisions, particularly parcel-to-loading station assignments, have the strongest standalone impact, while lower-tier decisions play an increasingly important role as facility size and delivery demand increase.

Building on these findings, the model could be extended to include additional reconfiguration decisions that enhance system flexibility, such as determining the number and placement of active loading stations or dynamically reallocating roll containers across destinations. A natural next step is to couple the planning layer with more detailed congestion- and routing-aware fleet control, for example by allowing robots to choose among multiple feasible routes and explicitly modeling the impact of congestion, queuing, and local traffic rules on travel times. Finally, variants in which the objective or constraints depend explicitly on parcel arrival times or sequences (e.g., makespan or time-based service guarantees) raise an interesting trade-off between model fidelity and computational effort, and offer an interesting direction for future work.

Declaration of the use of generative AI and AI-assisted technologies

During the preparation of this work the authors used ChatGPT in order to improve language and readability. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

Acknowledgment

Junsu Kim was supported by the National Research Foundation of Korea (NRF) through the Outstanding Graduate Students' Overseas Training Program of the BK21 FOUR (Brain Korea 21 Four) initiative. The authors gratefully acknowledge Professors Hosang Jung and Sang Yoon Lee of the Graduate School of Logistics, Inha University, for their support in securing this funding.

References

Ahuja, R. K., Magnantl, T. L., and Orlin, J. B. (1993). *Network flows: theory, algorithms, and applications*. Prentice-hall.

Alpan, G., Larbi, R., and Penz, B. (2011). A bounded dynamic programming approach to schedule operations in a cross docking platform. *Computers & Industrial Engineering*, 60(3):385–396.

Amazon (2022). See Inside Amazon's Robotics Headquarters in Suburban Boston. `https://www.aboutamazon.com/news/operations/amazon-robotics-headquarters-tour-westborough-massachusetts/`. Accessed: August 19, 2025.

Amazon (2025a). Amazon launches a new AI foundation model to power its robotic fleet and deploys its 1 millionth robot. `https://www.aboutamazon.com/news/operations/amazon-million-robots-ai-foundation-model/`. Accessed: August 19, 2025.

Amazon (2025b). Amazon's Robotics at the Fulfillment Center. `https://www.aboutamazon.com/news/operations/amazon-robotics-robots-fulfillment-center/`. Accessed: August 19, 2025.

Boysen, N., Briskorn, D., and Emde, S. (2017). Parts-to-picker based order processing in a rack-moving mobile robots environment. *European Journal of Operational Research*, 262(2):550–562.

Boysen, N., Briskorn, D., Fedtke, S., and Schmickerath, M. (2019). Automated sortation conveyors: A survey from an operational research perspective. *European Journal of Operational Research*, 276(3):796–815.

Boysen, N., Fedtke, S., and Schwerdfeger, S. (2021). Last-mile delivery concepts: a survey from an operational research perspective. *Or Spectrum*, 43(1):1–58.

Boysen, N., Schwerdfeger, S., and Ulmer, M. W. (2023). Robotized sorting systems: Large-scale scheduling under real-time conditions with limited lookahead. *European Journal of Operational Research*, 310(2):582–596.

Boysen, N., Stephan, K., and Schwerdfeger, S. (2024). Order consolidation in warehouses: The loop sorter scheduling problem. *European Journal of Operational Research*.

Chen, J. C., Anggrahini, D., and Chen, T.-L. (2024a). Current research and future challenges in parcel hub towards logistics 4.0: a systematic literature review from a decision-making perspective. *International Journal of Production Research*, pages 1–32.

Chen, Y., Xu, X., Zou, B., De Koster, R., and Gong, Y. (2024b). Assigning parcel destinations to drop-off points in a congested robotic sorting system. *Naval Research Logistics (NRL)*.

Crainic, T. G. (2000). Service network design in freight transportation. *European journal of operational research*, 122(2):272–288.

Dive, S. C. (2020). Amazon expands delivery station network to speed up logistics operations. `https://www.supplychaindive.com/news/amazon-delivery-station-logistics-operations-expansion/599062/`. Accessed: August 19, 2025.

Fang, Y., De Koster, R., Roy, D., and Yu, Y. (2025). Dynamic robot routing and destination assignment policies for robotic sorting systems. *Transportation Science*.

Fedtke, S. and Boysen, N. (2017). Layout planning of sortation conveyors in parcel distribution centers. *Transportation Science*, 51(1):3–18.

Figliozzi, M. A. (2017). Lifecycle modeling and assessment of unmanned aerial vehicles (drones) co2e emissions. *Transportation Research Part D: Transport and Environment*, 57:251–261.

Jarrah, A. I., Qi, X., and Bard, J. F. (2016). The destination-loader-door assignment problem for automated package sorting centers. *Transportation Science*, 50(4):1314–1336.

Jiang, X., Lee, K., and Pinedo, M. L. (2021). Ideal schedules in parallel machine settings. *European Journal of Operational Research*, 290(2):422–434.

Khir, R., Erera, A., and Toriello, A. (2021). Two-stage sort planning for express parcel delivery. *IISE Transactions*, 53(12):1353–1368.

Khir, R., Erera, A., and Toriello, A. (2023). Robust planning of sorting operations in express delivery systems. *European Journal of Operational Research*, 306(2):615–631.

Li, C., Wang, F., Gupta, J. N., and Chung, T. (2022). Scheduling identical parallel batch processing machines involving incompatible families with different job sizes and capacity constraints. *Computers & Industrial Engineering*, 169:108115.

McAree, P., Bodin, L., and Ball, M. (2002). Models for the design and analysis of a large package sort facility. *Networks: An International Journal*, 39(2):107–120.

Novoa, L. J., Jarrah, A. I., and Morton, D. P. (2018). Flow balancing with uncertain demand for automated package sorting centers. *Transportation Science*, 52(1):210–227.

Robotics, T. (2024). How Kmart Australia Cut Logistics Space by 60%. `https://blog.tompkinsrobotics.com/resources/case-studies/how-kmart-doubled-logistics-capacity/`. Accessed: August 19, 2025.

Rodrigue, J.-P. (2020). The distribution network of amazon and the footprint of freight digitalization. *Journal of transport geography*, 88:102825.

Shen, Y., McClosky, B., Durham, J. W., and Zavlanos, M. M. (2023). Multi-agent reinforcement learning for resource allocation in large-scale robotic warehouse sortation centers. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 7137–7143. IEEE.

Szeszák, B. M., Kerékjártó, I. G., Soltész, L., and Galambos, P. (2025). Industrial revolutions and automation: Tracing economic and social transformations of manufacturing. *Societies*, 15(4):88.

Tadumadze, G., Wenzel, J., Emde, S., Weidinger, F., and Elbert, R. (2023). Assigning orders and pods to picking stations in a multi-level robotic mobile fulfillment system. *Flexible Services and Manufacturing Journal*, 35(4):1038–1075.

Tan, Z., Li, H., and He, X. (2021). Optimizing parcel sorting process of vertical sorting system in e-commerce warehouse. *Advanced Engineering Informatics*, 48:101279.

Target (2024). How Target's Sortation Centers Deliver Joy for Guests and Efficiencies for Our Business. `https://corporate.target.com/press/fact-sheet/2023/02/sortation-centers/`. Accessed: August 19, 2025.

Werners, B. and Wülfing, T. (2010). Robust optimization of internal transports at a parcel sorting center operated by deutsche post world net. *European Journal of Operational Research*, 201(2):419–426.

Xu, X., Chen, Y., Zou, B., and Gong, Y. (2022). Assignment of parcels to loading stations in robotic sorting systems. *Transportation Research Part E: Logistics and Transportation Review*, 164:102808.

Yang, F., Davari, M., Wei, W., Hermans, B., and Leus, R. (2022). Scheduling a single parallel-batching machine with non-identical job sizes and incompatible job families. *European Journal of Operational Research*, 303(2):602–615.

Zhao, T., Lin, X., He, F., and Dai, H. (2024). Robotic sorting systems: Robot management and layout design optimization. *arXiv preprint arXiv:2404.04832*.

Zou, B., De Koster, R., Gong, Y., Xu, X., and Shen, G. (2021). Robotic sorting systems: Performance estimation and operating policies analysis. *Transportation Science*, 55(6):1430–1455.

Appendix A. System layout and travel time calculation

This appendix provides details on the layout and operational principles of the two-tier robotized sorting systems (T-RSS) used in our computational studies. The system consists of two tiers: (1) Upper tier: Robots transport parcels from loading stations to drop-off points; and (2) Lower tier: Robots deliver parcels from roll containers to shipping docks.

We derive the equations for travel times involved in these operations:

- t_i: travel time from the sort induction to loading station i by conveyor,
- $t_{i,j}$: travel time for a robot to transport a parcel from loading station i to drop-off point j,
- $t_{j,d}$: travel time for a robot to transport a roll container j to shipping dock d.

These components collectively determine the transportation cost κ_s for a parcel on path s. Parameters used in these derivations are summarized in Table A1 and are informed by prior studies (Boysen et al., 2023; Chen et al., 2024; Xu et al., 2022; Zou et al., 2021).

Table A1: Parameters for travel time calculation.

Parameter	**Description**
w_{ca}	Width of a cross-aisle (1.2m)
w_a	Width of an aisle (0.6m)
w_d	Width of a drop-off point (0.6m)
w_f	Distance from the sort induction to the main sortation floor (3m)
v_c	Velocity of the conveyor (2m/s)
v_m	Maximum velocity of the robot (2m/s)
a	Acceleration and deceleration rate (1m/s^2)
t_{90}	Time for a 90° turn (1s)

To establish the basis for robot travel time on both tiers, we derive the velocity and time relationship based on the travel distance, taking into account the effects of acceleration, deceleration, constant velocity phases, and turning conditions.

Let the travel distance be D, the robot's maximum velocity be v_m, and the acceleration/deceleration rate be a. Depending on whether the robot reaches its maximum velocity v_m, two cases arise (see Figure A1).

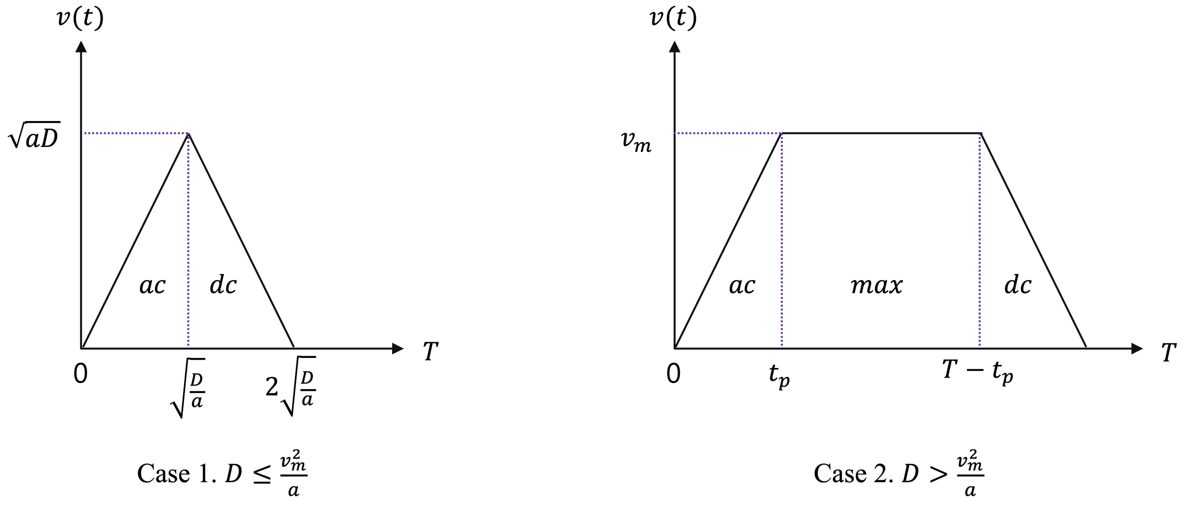


Figure A1: Velocity-time relationship based on travel distance.

Case 1: $D \leq \frac{v_m^2}{a}$. In this case, the robot does not reach v_m. The travel consists of two phases: (1) Acceleration ($[0, \sqrt{D/a}]$), and (2) Deceleration ($[\sqrt{D/a}, 2\sqrt{D/a}]$). Consequently, the total travel time is:

$$T = 2\sqrt{\frac{D}{a}}. \tag{A.1}$$

Case 2: $D > \frac{v_m^2}{a}$. Here, the robot reaches v_m and maintains it for a portion of the journey. The travel consists of three phases: 1. Acceleration ($[0, t_p]$) where $t_p = v_m/a$, 2. Constant velocity ($[t_p, T - t_p]$) where T includes the duration of constant-speed travel, 3. Deceleration ($[T - t_p, T]$). The total travel time is:

$$T = \frac{2v_m}{a} + \frac{D - \frac{v_m^2}{a}}{v_m}. \tag{A.2}$$

Based on these equations, we first analyze the upper tier, as illustrated in Figure A2.

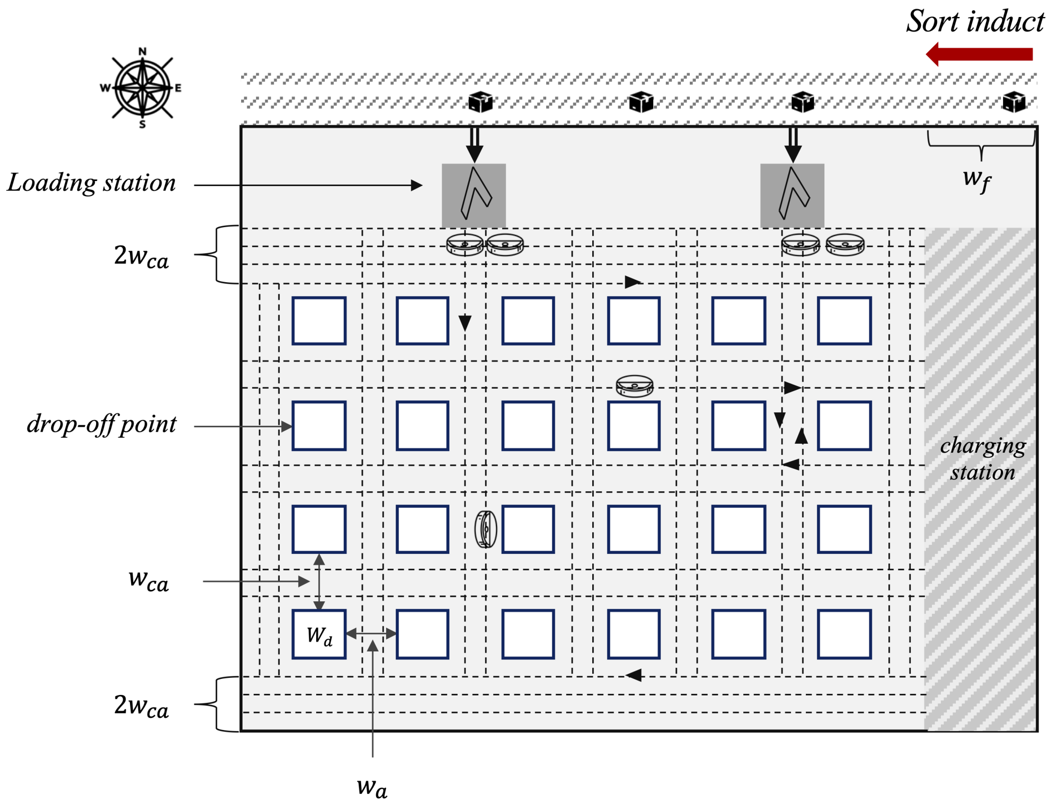


Figure A2: Upper-tier layout of the T-RSS (An example of small scale).

Although this process is not operated by robots, we begin by calculating the travel time from the sort induction to loading station i via a conveyor belt. The equations are defined as follows:

$$t_{i=1} = \frac{w_f + 2w_a + w_d}{v_c}. \tag{A.3}$$

$$t_{i\geq 2} = t_{i-1} + \frac{3w_d + 3w_a}{v_c}. \tag{A.4}$$

Here, w_a is the width of the aisle, w_d is the width of the drop-off point, and v_c is the conveyor's constant velocity. Note that the first loading station ($i = 1$) is located closest to the sort induction.

Next, for the sortation floor where robots transport parcels, we calculate the distance between loading station i and drop-off point j, denoted as $d_{i,j}$. This distance is used to determine the travel time $t_{i,j}$, calculated using the following cases based on Equations (A.1) and (A.2):

$$t_{i,j} = \begin{cases} 2\sqrt{\frac{d_{i,j}}{a}} + \Delta t_{90}, & \text{if } d_{i,j} \leq \frac{v_m^2}{a}. \\ \frac{2v_m}{a} + \frac{d_{i,j} - \frac{v_m^2}{a}}{v_m} + \Delta t_{90}, & \text{if } d_{i,j} > \frac{v_m^2}{a}. \end{cases} \tag{A.5}$$

Robot cornering is also included as an additional term. The turning condition, Δ, is defined as follows:

- If the drop-off point j is located in the first column at the bottom-left of loading station i, $\Delta = 0$ (no cornering occurs).
- If the drop-off point j is located beyond the first column at the bottom-left or anywhere at the bottom-right of loading station i, $\Delta = 1$ (cornering occurs once).

For detailed distance calculations and robot control rules in robotic sorting systems, please refer to Zou et al. (2021).

Similarly, the lower tier of the system involves the movement of robots between roll containers and shipping docks, as illustrated in Figure A3.

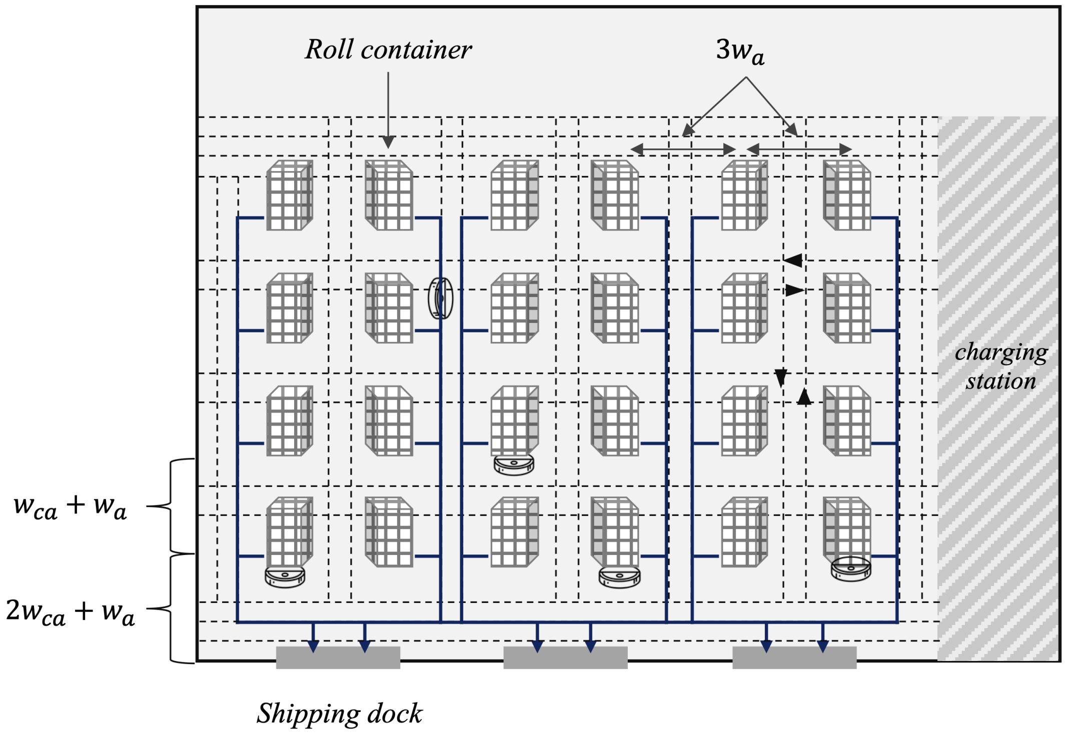


Figure A3: Lower-tier layout of the T-RSS (An example of small scale).

In the lower tier, the travel time for a roll container j to a shipping dock d, denoted as $t_{j,d}$, is calculated as:

$$t_{j,d} = \begin{cases} 2\sqrt{\frac{d_{j,d}}{a}} + 3t_{90}, & \text{if } d_{j,d} \leq \frac{v_m^2}{a}. \\ \frac{2v_m}{a} + \frac{d_{j,d} - \frac{v_m^2}{a}}{v_m} + 3t_{90}, & \text{if } d_{j,d} > \frac{v_m^2}{a}. \end{cases} \tag{A.6}$$

Here, $d_{j,d}$ represents the distance between roll container j and dock d, and t_{90} accounts for the time required for a 90° turn. We assume that three cornering maneuvers must occur, which can be adjusted based on the topology. For ease of distance calculation, the lower tier can be conceptualized as being primarily partitioned by shipping docks. This does not imply a zoning strategy that reduces the flexibility of optimization; rather, it reflects how the travel distance is affected based on whether the roll container is in the same column as the dock. For example, in the small-scale layout, two columns are connected to each shipping dock, resulting in three partitions. The following relationships define the distances:

1. **Between roll containers in the same column:**
   - The distance between the roll container in the first row (bottom) and the shipping dock is: $2w_{ca} + w_a$.
   - For subsequent rows, the distance increases by: $w_{ca} + w_a$.
2. **Between roll containers in different columns:**
   - For roll containers in the same row but different columns, the distance increases by: $3w_a$.
   - For roll containers in different columns but within the same partition, the additional distance is zero.

All the above explanations pertain to a small-sized layout for clarity and ease of understanding. For medium and large-sized layouts, the same logic applies with a greater number of nodes. Notably, varying distance parameters for different facility sizes can be seamlessly incorporated by inputting the distance matrix as a parameter into the optimization model.

Appendix B. Feasibility–recovery assignment model

This appendix describes an approach for testing whether an offline commodity plan can be lifted to the parcel level under the current available capacities, and for quantifying overflow when it cannot. The model is solved independently for each commodity (i.e., truck destination, k) and checks whether the prescribed path flows (h_k^s) can be realized exactly at the parcel level. When the integrality and balance conditions of Theorem 1(iii) hold and sufficient residual capacity is available, the model is feasible without a dummy path and directly recovers a valid parcel-to-path assignment. When these conditions fail, either because of structural imbalance or insufficient residual capacity, parcels assigned to the dummy path quantify the overflow that cannot be accommodated by the precomputed path set. Appendix B.2 provides a recovery procedure for this case.

B.1 Assignment model

Given an offline commodity plan with target flows h_k^s for $s \in S_k$, we recover a parcel-level assignment independently for each commodity k. For a fixed k, let P_k denote the parcels of truck k, $S_k^* \subseteq S_k$ the active paths selected by the offline plan, $h_k^s \in \mathbb{Z}_+$ the number of parcels prescribed on path $s \in S_k^*$, and κ_s the cost of assigning a parcel to path s.

The binary decision variable y_p^s equals 1 if parcel $p \in P_k$ is assigned to path $s \in S_k^* \cup \{\text{dummy}\}$ and 0 otherwise. The assignment model checks whether the path flows h_k^s are implementable at the parcel level given the current residual capacity state:

$$\begin{aligned}
\min \quad & \sum_{p \in P_k} \sum_{s \in S_k^*} \kappa_s \, y_p^s \; + \; M \sum_{p \in P_k} y_p^{\text{dummy}} \\
\text{s.t.} \quad & \sum_{s \in S_k^* \cup \{\text{dummy}\}} y_p^s = 1 && \forall p \in P_k, \\
& \sum_{p \in P_k} y_p^s = h_k^s && \forall s \in S_k^*, \\
& y_p^s \in \{0,1\} && \forall p \in P_k, \; \forall s \in S_k^* \cup \{\text{dummy}\}.
\end{aligned}$$

The dummy path (indexed "dummy") carries a large penalty $M \gg \max_{s\in S} \kappa_s$ and is used solely as an overflow indicator: any parcel assigned to the dummy path indicates that the prescribed flows h_k^s cannot be fully realized on S_k^*. Because the constraint matrix is totally unimodular, the model admits an integral optimal solution. If the integrality and balance conditions of Theorem 1(iii) hold, there exists an optimal solution with $y_p^{\text{dummy}} = 0$ for all $p \in P_k$ and the model yields a valid parcel-level lifting of the commodity plan. Otherwise, the total overflow

$$r_k := \sum_{p\in P_k} y_p^{\text{dummy}}$$

equals the number of parcels that cannot be assigned to S_k^* and must be handled by the recovery procedure in Appendix B.2.

B.2 Overflow recovery procedure

When overflow is detected, a second-stage recovery procedure is used to route the residual parcels. The idea is to (i) quantify the unmatched parcels using the assignment model above, (ii) re-optimize a smaller *Commodity*-SP problem on the residual demands, and (iii) lift the resulting flows using a second assignment step without a dummy path to verify.

Algorithm 1 Parcel-level recovery from a commodity plan

```
 1: **Input:** Offline flows $h_k^s$, path sets $S_k^*$, parcel sets $P_k$
 2: **Output:** Feasible parcel-to-path assignment $\{y_p^s\}$
    **Phase 1: Feasibility check and overflow detection**
 3: for $k \in K$ do
 4:     Solve the assignment model (with dummy path) for commodity $k$.
 5:     Compute overflow:
            $r_k \leftarrow \sum_{p\in P_k} y_p^{\text{dummy}}, \qquad P_k^{\text{ov}} \leftarrow \{p \in P_k : y_p^{\text{dummy}} = 1\}.$
 6: end for
 7: if $r_k = 0$ for all $k$ then
 8:     return the assignments $\{y_p^s\}$                ▷ Offline plan is fully liftable
 9: end if
    **Phase 2: Re-optimization for overflow demand**
10: Define residual demands $w_k^{\text{res}} \leftarrow r_k$.
11: Solve a reduced Commodity–SP on $w_k^{\text{res}}$ to obtain additional flows $g_k^s$ on a (possibly restricted) path
    set $S_k^{\text{res}}$.
    **Phase 3: Lifting of overflow flows**
12: for $k \in K$ with $r_k > 0$ do
13:     Solve a second assignment model for overflow parcels $P_k^{\text{ov}}$:
            $\sum_{p\in P_k^{\text{ov}}} y_p^s = g_k^s, \qquad \sum_{s\in S_k^{\text{res}}} y_p^s = 1,$
        with no dummy variable.
14: end for
15: return combined assignments from Phases 1 and 3.
```

Remark 1 (Aggregate vs. execution-level capacity)**.** *Parcel-level infeasibility may occur even when the offline plan satisfies aggregate capacity constraints, because execution resources (e.g., roll containers) are discrete and destination-specific. In this case, overflow in the assignment model reflects a lack of appropriately configured containers. The recovery procedure in Appendix B.2 redistributes overflow over additional paths but does not model container availability; any remaining overflow therefore indicates the need for additional execution capacity or intervention.*

Appendix C. Pseudocode for the RULE Algorithm

Algorithm 2 RULE: Sequential rule-based baseline

```
1: **Initialize:** for each $j \in J$: act($j$) $\leftarrow 0$, cap($j$) $\leftarrow 0$, truck($j$) $\leftarrow \varnothing$; fix random seed; $c \leftarrow 1$
2: **(P2L) Round-robin loading station assignment**
3: **for** parcels $p \in P$ in arrival order **do**
4:     $i \leftarrow I[\,((c-1) \bmod |I|) + 1\,]$;  assign $p \rightarrow i$;  $c \leftarrow c + 1$
5: **end for**
6: **(T2D) Random truck-to-dock matching (uniform, w/o replacement)**
7: draw a random permutation of $D$; assign each $k \in K$ to the next $d \in D$; set $a_{d,k} \leftarrow 1$
8: **(P2DP) Greedy parcel-to-drop-off by proximity with activation/capacity**
9: **for** parcels $p \in P$ in the same arrival order **do**
10:     let $i$ be the loading station of $p$; let $k$ be the truck (commodity) of $p$
11:     scan $J$ in two passes, both ordered by increasing $t_{i,j}$:
        (i) activated $j$ with truck($j$) $= k$ and cap($j$) $< \tau$; (ii) all remaining $j$
12:     **for** first feasible $j$ in this order **do**
13:         **if** act($j$) $= 1$ **and** truck($j$) $= k$ **and** cap($j$) $< \tau$ **then**
14:             assign $p \rightarrow j$; cap($j$) $\leftarrow$ cap($j$) $+ 1$; **break**
15:         **else if** act($j$) $= 0$ **then**
16:             act($j$) $\leftarrow 1$; truck($j$) $\leftarrow k$; cap($j$) $\leftarrow 1$; assign $p \rightarrow j$; **break**
17:         **end if**
18:     **end for**
19: **end for**
20: **(C2D) Induce container-to-dock from T2D**
21: **for** each activated $j$ **do**
22:     $k \leftarrow$ truck($j$);  $d \leftarrow d(k)$ from T2D;  register $j \rightarrow d$ (if first time)
23: **end for**
24: **return** all assignments
```

Appendix D. Performance comparison across all solution approaches

This appendix presents the computational results for all test instances generated by the *Parcel*-SP model, OTO (Exact), OTO (Heu), and the RULE approach, as detailed in Table A2. The computational performance is evaluated by comparing solution quality, measured by the objective function (total parcel travel time), and computation time required to generate a sorting plan. The table below reports the objective values (Obj, in seconds) and runtimes (CPU, in seconds) for each solution approach and layout size. Additionally, the percentage gap from the best-found solution (Gap, in percentage) is provided for OTO (Exact), OTO (Heu), and RULE relative to the *Parcel*-SP model. It should be noted that the *Parcel*-SP model failed to find a feasible solution for the large-sized layout within the 3,600-second time limit, leaving the best solution as that obtained by OTO (Exact). Accordingly, the Gap column for OTO (Exact) in the large-sized layout reports the gap reported by Gurobi.

Table A2: Performance comparison across layout sizes and solution approahces.

Layout Size	# parcels	*Parcel*-**SP**		**OTO (Exact)**			**OTO (Heu)**			**RULE**		
		Obj(s)	CPU(s)	Obj(s)	Gap(%)	CPU(s)	Obj(s)	Gap(%)	CPU(s)	Obj(s)	Gap(%)	CPU(s)
	660	10863.30	5.50	10863.30	0.00	0.09	10935.30	0.66	0.14	12483.51	14.91	0.01
	314	5010.58	1.33	5010.58	0.00	0.06	5049.40	0.77	0.14	5834.54	16.44	0.01
	561	9207.20	26.27	9207.20	0.00	0.09	9296.00	0.96	0.14	10869.79	18.06	0.01
	654	10773.20	16.22	10773.20	0.00	0.12	10782.88	0.09	0.15	12519.47	16.21	0.01
	424	6810.20	1.76	6810.20	0.00	0.11	6870.40	0.88	0.14	8222.68	20.74	0.01
	699	11518.10	5.66	11518.10	0.00	0.08	11563.69	0.40	0.17	13382.78	16.19	0.01
	401	6442.10	2.63	6442.10	0.00	0.09	6482.20	0.62	0.13	7395.81	14.80	0.01
Small	501	8163.76	33.75	8163.76	0.00	0.12	8210.20	0.57	0.14	9549.70	16.98	0.01
	461	7484.56	18.67	7484.56	0.00	0.10	7519.06	0.46	0.14	8684.09	16.03	0.01
	402	6497.52	1.94	6497.52	0.00	0.07	6521.44	0.37	0.14	7599.53	16.96	0.01
	443	7131.80	2.05	7131.80	0.00	0.09	7228.70	1.36	0.14	8377.14	17.46	0.01
	773	12904.20	16.98	12904.20	0.00	0.11	12955.20	0.40	0.16	14896.50	15.44	0.01
	408	6608.40	17.38	6608.40	0.00	0.11	6620.40	0.18	0.14	7722.95	16.87	0.01
	530	8648.02	11.12	8648.02	0.00	0.08	8665.20	0.20	0.14	10122.67	17.05	0.01
	676	11165.54	5.95	11165.54	0.00	1.02	11177.18	0.10	0.15	13094.68	17.28	0.01
Avg	527.13	8615.23	11.15	8615.23	0.00	0.16	8658.48	0.54	0.14	10050.39	16.76	0.01
	1001	17517.24	108.43	17517.24	0.00	1.44	17641.10	0.71	2.89	23402.72	33.60	0.02
	603	10064.46	82.22	10064.46	0.00	1.18	10086.10	0.22	2.19	14475.66	43.83	0.01
	998	17048.00	83.45	17048.00	0.00	1.19	17300.60	1.48	2.27	24054.63	41.10	0.01
	1297	23394.86	140.99	23394.86	0.00	2.01	23477.16	0.35	2.34	32157.02	37.45	0.02
	840	14171.39	95.29	14171.39	0.00	1.51	14311.48	0.99	2.09	19516.89	37.72	0.01
	1178	20719.79	153.02	20719.79	0.00	1.88	20897.39	0.86	2.41	28261.15	36.40	0.02
	1314	23504.59	182.91	23504.59	0.00	1.93	23657.89	0.65	2.45	31723.98	34.97	0.02
Medium	1030	17732.12	112.10	17732.12	0.00	1.43	17867.16	0.76	2.57	26045.69	46.88	0.02
	1263	22452.12	130.48	22452.12	0.00	1.67	22652.78	0.89	2.59	30830.47	37.32	0.02
	798	13578.20	72.45	13578.20	0.00	1.09	13715.24	1.01	2.15	18946.28	39.53	0.02
	745	12486.22	722.33	12486.22	0.00	1.47	12519.51	0.27	2.03	18495.76	48.13	0.01
	1016	17607.60	127.46	17607.60	0.00	1.54	17700.11	0.53	2.31	21187.03	20.33	0.01
	815	13935.32	94.38	13935.32	0.00	1.04	13981.00	0.33	2.95	19693.37	41.32	0.01
	919	15945.08	104.97	15945.08	0.00	1.47	15998.23	0.33	2.23	22063.86	38.37	0.01
	1357	24456.48	257.90	24456.48	0.00	1.72	24559.53	0.42	2.23	32970.49	34.81	0.02
Avg	1011.60	17640.90	164.56	17640.90	0.00	1.50	17757.69	0.65	2.38	24255.00	38.12	0.02
	2535	-	-	55325.96	0.10	3600.00	55887.42	1.01	81.86	84311.89	52.39	0.08
	2248	-	-	47738.56	0.07	3600.00	48603.86	1.81	91.06	74713.23	56.50	0.06
	2474	-	-	53177.98	0.13	3600.00	54108.65	1.75	85.76	80632.42	51.63	0.07
	2426	-	-	53059.92	0.05	3600.00	53535.18	0.90	94.02	80562.09	51.83	0.07
	3125	-	-	72154.02	0.10	3600.00	73406.40	1.74	94.11	103505.95	43.45	0.08
	2945	-	-	67211.50	0.10	3600.00	68401.10	1.77	87.12	97372.36	44.87	0.07
	1869	-	-	38956.14	0.02	3600.00	39563.00	1.56	75.69	61212.82	57.13	0.05
Large	2449	-	-	52751.10	0.14	3600.00	53710.66	1.82	69.93	80832.39	53.23	0.09
	2477	-	-	54236.08	0.25	3600.00	54238.80	0.01	83.82	81020.74	49.39	0.09
	2606	-	-	57642.58	0.24	3600.00	58386.09	1.29	75.90	83441.02	44.76	0.09
	2463	-	-	53600.36	0.11	3600.00	53908.14	0.57	65.88	84190.46	57.07	0.09
	2168	-	-	46392.18	0.15	3600.00	47160.50	1.66	65.45	74015.52	59.54	0.07
	2549	-	-	55159.24	0.07	3600.00	57291.20	3.87	78.78	83594.75	51.55	0.09
	2155	-	-	45865.46	0.07	3600.00	46288.46	0.92	80.23	73728.84	60.75	0.08
	2892	-	-	64767.56	0.20	3600.00	65561.80	1.23	85.08	96958.88	49.70	0.09
Avg	2492.07	-	-	54535.91	0.12	3600.00	55336.75	1.46	80.98	82672.89	52.25	0.08

Appendix E. Performance comparison across different integration cases

This appendix outlines the testing of seven integration cases and their corresponding numerical analyses. The fully integrated case, JOINT (1, 2, 3, 4), which incorporates all four decisions simultaneously, was evaluated alongside two three-decision cases: JOINT (1, 2, 3) and JOINT (2, 3, 4). Additionally, three two-decision cases—JOINT (1, 2), JOINT (2, 3), and JOINT (3, 4)—were analyzed, as well as the fully independent scenario, JOINT (0).

For non-integrated decisions, they are solved independently, fixed in the optimization model, and the remaining decisions are solved sequentially. For example, in JOINT (1, 2, 3), decision (4) is determined using a predefined rule, fixed in the optimization model to reduce the search space, and the remaining decisions are solved simultaneously.

Only objective values (Obj, in seconds) are reported for JOINT (1, 2, 3, 4), which serves as the benchmark. For the JOINT variants, we report the percentage gap (Gap, %) relative to JOINT (1, 2, 3, 4), indicating the increase in travel time, as well as the computational runtime (CPU, in seconds). It is worth noting that the *Parcel*-SP model failed to find a feasible solution for the fully integrated case, JOINT (1, 2, 3, 4). However, solutions were obtained for partially integrated cases, with significantly shorter computation times due to the reduced search space, enabling result comparison with OTO (Exact) in large-sized layouts. Nevertheless, some instances at this problem scale still did not achieve optimal solutions within the 3,600-second time limit. In such cases, Gap_g values, representing the optimality gap reported by Gurobi at the time limit, are provided.

Accordingly, results are categorized into small- and medium-sized layouts (summarized in Table A3) and large-sized layouts (detailed in Table A4).

Table A3: Performance comparison across different integration cases (Small & Medium).

Layout Size	**# parcels**	**JOINT (1,2,3,4)**	**JOINT (1,2,3)**		**JOINT (2,3,4)**		**JOINT (1,2)**		**JOINT (2,3)**		**JOINT (3,4)**		**JOINT (0)**	
		Obj(s)	Gap(%)	CPU(s)	Gap(%)	CPU(s)	Gap(%)	CPU(s)	Gap(%)	CPU(s)	Gap(%)	CPU(s)	Gap(%)	CPU(s)
Small	660	10863.30	0.64	2.24	5.86	18	5.35	1.15	6.10	4.55	14.55	1.06	14.91	0.01
	314	5010.58	0.00	1.04	7.19	2.53	4.01	0.53	8.60	1.02	12.45	1.51	16.44	0.01
	561	9207.20	3.80	1.88	5.87	14.65	7.08	0.90	6.00	2.59	12.90	1.99	18.06	0.01
	654	10773.20	0.00	2.47	5.86	7.03	5.49	1.16	5.86	1.52	12.45	1.45	16.21	0.01
	424	6810.20	1.93	1.38	6.60	3.30	5.22	0.73	12.47	1.08	15.82	1.56	20.74	0.01
	699	11518.10	1.69	2.38	5.69	29.46	6.16	1.20	5.86	2.13	15.11	1.34	16.19	0.01
	401	6442.10	0.00	1.27	6.91	3.32	5.12	0.66	8.36	1.11	14.80	1.69	14.80	0.01
	501	8163.76	3.31	1.64	6.02	12.82	6.88	0.82	6.02	3.00	14.36	2.55	16.98	0.01
	461	7484.56	0.62	1.66	6.35	8.34	4.98	0.88	6.40	1.80	15.67	2.24	16.03	0.01
	402	6497.52	0.64	1.50	6.42	6.30	3.29	0.78	6.77	1.38	12.17	1.61	16.96	0.01
	443	7131.80	5.46	1.60	6.78	3.56	8.09	0.76	7.54	1.07	11.30	2.34	17.46	0.01
	773	12904.20	0.00	4.11	5.06	34.73	5.62	1.82	5.28	5.79	12.87	1.48	15.44	0.01
	408	6608.40	0.46	1.30	6.42	4.68	4.73	0.67	7.38	1.66	13.75	2.24	16.87	0.01
	530	8648.02	1.52	1.79	6.30	9.50	4.80	1.19	6.68	1.72	13.20	1.61	17.05	0.01
	676	11165.54	0.11	2.25	5.72	6.23	5.41	1.14	5.78	5.35	14.19	1.91	17.28	0.01
	Avg	8615.23	1.34	1.90	6.20	10.96	5.48	0.96	7.01	2.38	13.71	1.77	16.76	0.01
Medium	1001	17517.24	6.70	7.66	15.39	145.34	11.86	7.34	16.28	14.08	31.37	2.87	33.60	0.02
	603	10064.46	7.80	7.10	19.97	74.09	13.44	7.35	23.39	10.88	31.21	2.22	43.80	0.01
	998	17048.00	15.80	8.19	17.64	131.91	19.39	6.05	20.80	12.24	33.99	2.27	41.10	0.01
	1297	23394.86	1.90	10.99	13.08	486.11	10.02	9.99	14.09	17.44	27.66	3.73	37.50	0.02
	840	14171.39	15.90	7.66	18.55	55.17	19.35	7.34	22.00	18.45	26.99	2.57	37.70	0.01
	1178	20719.79	0.70	7.45	15.21	397.11	13.50	7.67	17.28	20.25	28.59	2.48	36.40	0.02
	1314	23504.59	1.60	8.58	14.12	280.39	11.57	7.93	15.48	19.87	26.64	1.89	35.00	0.02
	1030	17732.12	5.80	7.67	16.66	106.81	13.46	7.63	19.70	14.18	30.26	2.11	46.90	0.02
	1263	22452.12	4.40	9.61	14.20	393.99	11.76	9.01	16.06	20.42	32.42	1.79	37.30	0.02
	798	13578.20	5.10	9.30	18.34	238.01	11.33	9.98	19.11	11.09	34.75	2.29	39.50	0.02
	745	12486.22	8.50	9.33	12.60	495.30	17.18	9.60	24.31	10.23	37.50	1.32	48.10	0.01
	1016	17607.60	5.50	8.98	12.24	100.98	12.19	9.36	18.12	10.03	18.53	2.38	20.30	0.01
	815	13935.32	1.50	7.09	12.47	88.35	11.71	7.04	20.14	10.22	33.52	1.88	41.30	0.01
	919	15945.08	7.00	6.23	16.52	159.81	12.46	6.66	18.57	17.44	30.57	1.65	38.40	0.01
	1357	24456.48	2.20	10.66	13.29	207.63	11.07	9.21	14.52	25.71	29.18	2.54	34.80	0.02
	Avg	17640.90	6.04	8.43	15.35	224.07	13.35	8.14	18.66	15.50	30.21	2.27	38.12	0.02

Table A4: Performance comparison across different integration cases (Large).

Layout Size	**# Parcels**	**JOINT (1,2,3,4)**	**JOINT (1,2,3)**			**JOINT (2,3,4)**			**JOINT (1,2)**		**JOINT (2,3)**		**JOINT (3,4)**		**JOINT (0)**	
		Obj(s)	Gap(%)	Gap_g(%)	CPU(s)	Gap(%)	Gap_g(%)	CPU(s)	Gap(%)	CPU(s)	Gap(%)	CPU(s)	Gap(%)	CPU(s)	Gap(%)	CPU(s)
Large	2535	55325.96	8.47	-	2255.76	20.18	0.05	3600	26.34	2054.30	29.40	2105.38	40.69	41.92	52.39	0.08
	2248	47738.56	9.48	-	2036.20	20.93	0.10	3600	26.99	1948.15	32.50	1220.31	38.74	35.26	56.50	0.06
	2474	53177.98	6.53	-	2047.84	19.79	0.10	3600	16.41	2031.89	23.66	1222.75	41.21	38.47	51.63	0.07
	2426	53059.92	7.28	-	2024.56	20.66	0.09	3600	23.37	2047.63	27.06	1241.12	40.25	38.35	51.83	0.07
	3125	72154.02	6.67	0.20	3600.00	17.18	0.13	3600	16.27	3298.37	19.53	2577.01	33.73	57.89	43.45	0.08
	2945	67211.50	5.61	-	2623.44	17.79	0.03	3600	17.45	1923.51	24.18	3323.02	37.18	52.50	44.87	0.07
	1869	38956.14	10.08	-	1830.73	21.83	0.02	3600	21.30	1235.84	31.63	1202.72	40.73	23.11	57.13	0.05
	2449	52751.10	7.19	-	2810.06	18.58	0.09	3600	17.73	1823.37	20.07	1980.93	37.85	40.75	53.23	0.09
	2477	54236.08	7.60	-	2044.20	17.77	0.10	3600	29.45	1909.36	30.46	2320.26	33.40	36.98	49.39	0.09
	2606	57642.58	7.40	-	2674.13	19.56	0.07	3600	16.08	2071.93	25.94	1997.61	33.56	41.32	44.76	0.09
	2463	53600.36	6.38	-	1884.26	19.58	0.05	3600	22.06	1548.55	27.26	1552.60	41.81	42.75	57.07	0.09
	2168	46392.18	6.77	-	1860.74	20.83	0.03	3600	22.58	1456.59	29.41	1360.27	39.82	29.08	59.54	0.07
	2549	55159.24	11.52	-	850.30	18.09	0.07	3600	27.48	1334.25	32.36	2217.30	38.76	39.01	51.55	0.09
	2155	45865.46	10.19	0.15	3600.00	19.12	0.02	3600	17.98	1637.88	32.97	1987.21	37.63	28.92	60.75	0.08
	2892	64767.56	8.01	-	1923.45	18.06	0.09	3600	22.51	1866.55	24.22	2632.50	36.60	50.04	49.70	0.09
	Avg	54535.91	7.95	0.18	2271.04	19.33	0.07	3600	21.60	1879.21	27.38	1929.40	38.13	39.76	52.25	0.08

References

Boysen, N., Schwerdfeger, S., and Ulmer, M. W. (2023). Robotized sorting systems: Large-scale scheduling under real-time conditions with limited lookahead. *European Journal of Operational Research*, 310(2):582–596.

Chen, Y., Xu, X., Zou, B., De Koster, R., and Gong, Y. (2024). Assigning parcel destinations to drop-off points in a congested robotic sorting system. *Naval Research Logistics (NRL)*.

Xu, X., Chen, Y., Zou, B., and Gong, Y. (2022). Assignment of parcels to loading stations in robotic sorting systems. *Transportation Research Part E: Logistics and Transportation Review*, 164:102808.

Zou, B., De Koster, R., Gong, Y., Xu, X., and Shen, G. (2021). Robotic sorting systems: Performance estimation and operating policies analysis. *Transportation Science*, 55(6):1430–1455.