Use statistical analysis to approximate integrated order batching problem

Sen Xue and Chuanhou Gao
School of Mathematical Sciences, Zhejiang University, Hangzhou, People’s Republic of China

ABSTRACT
This paper highlights the tight relationship between the picking and packing processes in warehouse management and the need to consider them as an integrated problem. The study describes and models this integrated problem as a mixed-integer programming model, to optimise overall labour costs by determining the assignment of the subsets of orders, i.e. batches, for picking and packing. To address the issue of model complexity, the paper presents a statistical-based framework for generating approximate models and selecting the optimal one through examination. Based on the examination results, a pair-swapping heuristic is additionally proposed to be combined as a hybrid algorithm. Numerical experiments based on a real-world case demonstrate the effectiveness of the framework-proposed and selected hybrid algorithm by comparison with other framework-proposed approximate models, a solver, and existing heuristics. Our findings indicate that the combined usage of integrated picking and packing processes planning and the hybrid algorithm proposed and selected within the statistical-based framework can effectively reduce the cost of warehouse management.

1. Introduction

Warehouse management encompasses a series of non-independent product handling operations, including receiving, storage, order picking, packing, and shipping (Hompel and Schmidt 2007). The order-picking process, defined as retrieving products from storage locations, is the most labour-intensive and costly procedure. According to Bartholdi and Hackman (2014), the picking consumes around 55% of the total warehouse cost. Following close to the allocation of orders, the packing process, which also involves manual labour, concerns the sortation and packaging of orders. This operation sequence indicates that, in contrast to receiving, storage, and shipping which are more closely related to logistics and warehouse design, packing is more strongly correlated to the picking process. Thus the idea of considering them jointly should be taken into account. As Ackerman (2012) mentioned, considering picking and packing together would provide a more accurate description of warehouse operations and avoid extra storage buffers. The recent paper by Zhong et al. (2022) has numerically demonstrated the efficacy of addressing an integrated picking and packing problem in comparison to separately considering the picking problem.

Order batching is a key strategy in warehouse management that partition orders into groupings when there exists a capacity constraint in picking facilities. The order batching problem (OBP) considers the optimal batch assignment that minimises the total cost, which is mostly defined as the travel distance of item pickers in the picking process. A significant amount of research has investigated the modelling and solving method of OBP in picking alone. However, a very limited number of researchers investigated the OBP considering picking and packing jointly. The effectiveness, modelling, and solving method of this approach in different cases are not fully studied, which relatively leaves a gap for our study.

In this paper, we give an order batching problem integrated with the packing process, based on a warehouse case of one of China’s largest logistics companies. In this case, we find that the batch assignment can directly affect the performance of the packing process by determining how many unique items, i.e. the types of items, in each batch are to be sorted. This situation motivates us to propose an integrated problem. Further, the integrated problem presents a challenge of a more complex model with more constraints and variables which makes it impractical to be solved by a solver within a reasonable time frame, especially for middle or large instances. Meanwhile, it is observed that the problem needs to be solved repeatedly, with many parameters remaining unchanged, such as item locations and item types. The question arises
as to whether past data can be utilised, and if the distribution of solutions can be characterised for the construction of a new algorithm. These two questions serve as the starting point of our investigation of a new statistical solving method.

1.1. Literature review

1.1.1. Picking and packing as an integrated problem

Over the last decade, a new trend has emerged that attempts to consider the operation parts jointly. This includes order batching and routine problem (Chen et al. 2015; Valle, Beasley, and Da Cunha 2017; Van Gils et al. 2018), storage and picking problem (Calzavara et al. 2019; Jiang et al. 2021; Tappia et al. 2019; Wang, Zhang, and Fan 2020). See reviews by Esra and Nil (2022) and Cergibozan and Tasan (2019) on the integrated problems and their variants. However, a rather limited number of publications consider the picking and packing process jointly.

Several researchers presented comprehensive case studies with solving methods for joint order picking and packing problems. Shiau and Lee (2010) provided a case study in a tea factory, where customer orders have different sizes. Thus, considering an efficient picking sequence and product loading locations in each container would result in the reduction of operation time and boxing space in packaging. For this aim, a hybrid algorithm for picking and packing operations to generate a picking sequence is proposed, which includes a linear programming model and heuristics. De Koster, Le-Duc, and Zaerpour (2012) described a warehouse with pick-and-sort systems and zones, which are divided areas for parallel picking operations. The study illustrated that an excessive number of zones would reduce picking time, but concurrently increases the packing time, and vice versa as an order must be picked completely before packing. The study then proposed an integer-programming model to determine the optimal number of zones for total operation time. Zhong et al. (2022) developed a mixed-integer non-linear programming model for order picking and packing processes in e-commerce warehouses. The study investigated the improvement brought by modelling the integrated problem compared with a nonintegrated way and proved it numerically by simulation experiments.

Additionally, several studies have demonstrated that under the pick-and-sort policy, the sorting process between picking and packing could be improved, resulting in reduced idle time during packing. Gallien and Weber (2010) initially proposed a model to forecast the movement patterns of goods within a warehouse (i.e. sorter and packer usage) when employing a waveless approach. Subsequently, the model was used to control the main parameters of this strategy, aiming to achieve the highest possible throughput. Simulation experiments were conducted to prove the efficiency. Boysen, Fedtke, and Weidinger (2018) devised an elementary optimisation problem to shorten the packing time by minimising the spread of orders in the release sequence from the automated storage and retrieval system. Further, Boysen, Stephan, and Weidinger (2019) reduced the idle time of packers by determining the bin sequence.

All the studies above are based on real-world cases and are deeply related to problem characteristics. Given this current state of research, providing a more diverse range of case-based problem formulations is beneficial. Despite that these studies have effectively addressed the problem, they are limited in comparison to traditional research on order batching, as few of them focus on determining the batch assignment directly in this integrated problem and the scale of the problem is relatively small. Furthermore, there is a lack of new algorithms proposed in these studies, with the majority of the research utilising mixed integer programming as the primary method of solution.

1.1.2. Solving methods of order batching problem

The order batching and order packing problems in this study still share main features with classic order batching problems as they all consider the batch assignment of orders. The warehouse environment of the OBP can be separated into two distinct categories: picker-to-parts system and part-to-picker system. In a picker-to-parts system, the item picker travels to designated storage locations to gather the required items, whereas, in a part-to-picker system, the items are retrieved and brought to the item picker automatically.

Until now, the majority of OBP solution approaches have been heuristics and metaheuristics, which are widely employed in practice. Only a few works attempt to solve OBP to optimality. There is also a limited number of rule-based methods. One main reason is that the industry emphasises the timeliness of an algorithm much more than the accuracy of the solution. The computing time of the B&B algorithm makes it impractical for medium or large-scale order batching problems.

Heuristic algorithms consider defined rules for assigning orders. Gibson and Sharp (1992) introduced the First-Come-First-Served (FCFS) rule for OBP, which is the most straightforward. Elsayed (1981) and Elsayed and Unal (1989) firstly developed the seed algorithm. The seed algorithm uses selection rules to choose the starting order for each batch, then uses addition rules to add the remaining orders. The rules are formulated based on the problem characteristics. The saving algorithm, which was initially proposed by Clarke and Wright (1964) for the Vehicle Routing Problem, has evolved into many OBP
variants, among which C&W(i) and C&W(ii) are mostly mentioned.

Metaheuristics are mostly common algorithms modified to OBP. Albareda-Sambola et al. (2009) developed a variable neighbourhood search (VNS) based approach with four different warehouse configurations. Henn et al. (2010) presented two methods, an iterated local search (ILS) and a rank-based ant system. Henn and Wäscher (2012) proposed tabu search (TS) to minimise the total length of picking routes. Žulj and Kramer and Schneider (2018) proposed a hybrid metaheuristic based on adaptive large neighbourhood search and tabu search, named ALNS/TS. This method showed more efficiency on larger OBP instances compared with the method of Henn and Wäscher (2012). Recently, Van Gils et al. (2019) gave an ILS algorithm to solve the problem that integrates order batching, picker routing, and picker scheduling, which is an example of metaheuristics used on integrated order batching problems. Efforts were also made on other metaheuristics like the genetic algorithm (GA) (Chen et al. 2015; Jiang et al. 2022; Zare Mehrjerdi, Alipour, and Mostafaei-Pour 2018) and ant colony optimisation (ACO) (Chen et al. 2016). Hybrid metaheuristic approaches were also investigated by Zhang et al. (2017) and Cheng et al. (2015).

For exact methods, Tang et al. (2011) proposed a Lagrangian relaxation and column generation-based mixed-integer programming (MIP) approach. Following that, Muter and Öncan (2015) developed a column generation-based exact method. The disadvantage of exact methods is that they cannot be extended to larger application instances (e.g. more than 100 orders). However, in this paper, we improve upon this situation by introducing an approximation MILP approach.

Several rule-based methods were proposed. Chen and Wu (2005) produced the first method combining data mining and MIP for OBP. They introduced the association rule mining to OBP and developed a MIP for maximising order association in a batch and indirectly obtaining high-quality solutions with small total distances. Ho and Tseng (2006) proposed several batching methods established on seed-order selection rules (SOSR) and accompanying-order selection rules (AOSR). A subsequent study by Ho, Su, and Shi (2008) on this topic gave new rules and interactions between SOSR and AOSR. Ming-Huang Chiang, Lin, and Chen (2014) proposed a new association measure based on association rule mining, which drives two heuristics presented by the authors: the modified class-based heuristic (MCBH) and the association seed-based heuristic (ASBH). These two heuristics showed an improvement in efficiency compared with traditional approaches in numerical experiments. Aboelfotoh, Singh, and Suer (2019) proposed a heuristic suitable for larger instances, which aims to put similar orders to batch together based on their item locations.

1.1.3. Statistical and learning approximation methods for integer programming

Statistical studies focus on estimating the optimal value of a given optimisation problem. For studies related to routing or logistics, see reviews of Franceschetti, Jabali, and Laporte (2017) and Choi and Schonfeld (2021). The studies commence with the Euclidean travelling salesman problem, where $n$ cities are placed independently and randomly in a d-dimensional hypercube. The Euclidean metric is employed to define the distances between cities. Define $S_n$ as a random variable of the optimal tour length for $n$ cities. Beardwood, Halton, and Hammersley (1959) guessed that $S_n$ follows a normal distribution based on the law of large numbers but cannot prove it, which is still an open problem today. In computation experiments, Applegate et al. (2007) generated 10,000 instances of 1000 cities and showed that $S_n$ appears to follow a normal distribution. Beardwood, Halton, and Hammersley (1959) also proposed an approximation formula for $S_n$, known as the BHH formula, which can be written as

$$\lim_{n \to \infty} \frac{S_n}{\sqrt{n}} = \gamma$$

with probability 1.

However, the value of $\gamma$ is unknown. Research efforts have been made to determine accurate bounds of $\gamma$, among which the latest is the study by Arlotto and Steele (2016). Furthermore, studies (Dubhashi and Panconesi 2009; Rhee and Talagrand 1989; Steele 1981) proposed several the bounds on $P\{|S_n - E[S_n]| > t\}$ for a given $t$. As for extension works, approximation formulas concerning the Vehicle Routing Problem (VRP) have also been proposed by Figliozzi (2007, 2008). In addition to the formula-based estimation, recent studies also employ a variety of statistical methods. Nicola, Vetschera, and Dragomir (2019) used regression-based estimation models to estimate the travel distance in the travelling salesman problem (TSP), VRP, and its variants. Merchan and Winkenbach (2019) extended a data-driven approach to continuum approximation-based methods, aiming to predict urban route distances. Akkerman and Mes (2022) applied several regression models for TSP and VRP cost estimation and proved their accuracy.

In recent years, various learning methods have been applied to approximately solving general integer programming problems. The motivation of such studies is to use learning methods to replace some existing heavy computation steps in a fast approximation way (Bengio, Lodi, and Prouvost 2021). The training process is referred to as imitation learning, during which time-consuming
but accurate steps are executed and the results are used as a training set for machine learning. Gasse et al. (2019) proposed a graph convolutional neural network model to imitate the strong branching strategy, which would compute the bound improvement for each candidate branching node during the branch and bound process. In addition, several other studies have been proposed to address the issue of branching (Alvarez, Louveaux, and Wehenkel 2017; Etcheve et al. 2020; Nair et al. 2020). Furthermore, the replication efforts also encompass the cutting plane selection (Huang et al. 2022; Paulus et al. 2022; Tang, Agrawal, and Faenza 2020) and column selection in the Column Generation Algorithm (Morabit, Desaulniers, and Lodi 2021).

The statistical methods, while effectively determining the value of the optimal solution, are not directly applicable to the solution process. On the other hand, while the learning methods are capable of producing high-quality solutions in a relatively short amount of time, the black-box nature of these methods presents challenges in regard to performance analysis (Fan et al. 2021).

1.2. Contributions and paper structure

Due to the practical motivations behind the real-world problem, existing gaps in the research, and limitations of conventional statistical methods, we propose an effective and non-black-box approach as one of our contributions. Our contributions are as follows:

1. We formulate a new variant of the order batching and order packing problem under different routeing strategies.
2. We present a statistical-based framework for the generation of approximate MIP models and provide accompanying techniques for the estimation of the model’s utility.
3. We generate three approximate MIP models within the statistical-based framework and examine them for selection. Moreover, we develop a problem-adapted order pair-swapping heuristic based on the empirical relationship found by examination.

This paper is organised as follows. Section 2 presents a description and model formulations of the OBOPP. Section 3 introduces a statistics-based framework to maintain an approximation MILP for OBOPP. Section 4 shows generation rules for the approximation MIP. Section 5 gives a pair-swapping heuristic for reinforcement. Section 6 examines our methods with the exact B&B algorithm, classic heuristics of OBP, and their variants. Finally, Section 7 concludes the study and gives future research directions.

2. The order batching and order packing problem

2.1. Problem description

In this subsection, we describe the order batching and order packing problem (OBOPP). A warehouse is composed of parallel aisles and a packing area with a fixed number of packers. The packing area is in the middle right of the last aisle (See Figure 1). This layout is similar to De Koster, Le-Duc, and Zaerpoor (2012). In this study, the aisles are lined in length-descending order, from the left to the right side of the warehouse. This is a small traditional layout modification that arranges frequently ordered item classes to be stored on short aisles, leading to generally lower picking route distances. It is vital to note that our model considering this specialty can also be compatible with a traditional layout. For the relationship between orders and items, we follow the environment set by De Koster, Van der Poort, and Wolters (1999). Each
item has a unique storage location. An order, which represents a requirement of our customers, contains one or more items. The same items may appear in multiple orders at the same time.

The OBOPP can be described as a problem of finding a batching combination for the least cost of travelling distance and packing workload. At the beginning of each batch, the item-pickers depart from the depot point, travel down the aisles that index the locations of items, and collect the items. Then, the item-pickers carry the collected items to the packing point, where they are registered and sorted into orders. The orders are then packaged for delivery (See Figure 1 for illustration).

In this pipeline-style process, which combines routeing, sorting, and packaging, maintaining a stable workload, particularly at the final packaging point, is of major importance. This requirement is primarily driven by the number of personnel available at the packaging point, who can only process a fixed number of orders at any given time. As a result, an unstable workload can lead to increased waiting time for packaging personnel, ultimately resulting in increased costs. Additionally, demands from the delivery side also necessitate a steady loading quantity according to time. A simple but effective solution in practice is to set a fixed number of batch orders and an integer multiple of the number of pickers while maximising the number of them available, and ensuring that it falls within the capacity of pickers. Practically, a suitable setting will minimise the waiting time for both pickers and packers, making it within tolerable limits and omittable.

The objective function for modelling the cost of this entire process is based on the labour cost incurred in executing the various operations. Specifically, the operations in question are routeing, sorting, and packaging. The calculation of the cost associated with these operations is performed by dividing the workload by the execution speed and subsequently multiplying the result by the hourly wage for the corresponding job. In practical terms, the workload for routeing is quantified in terms of the distance travelled, denoted as \( DT \). The workload for sorting is represented by the number of unique items, denoted as \( NU \), where the number of unique items refers to the count of distinct item types, as in practice, sorting a single type of item into the desired orders requires a fixed amount of time. The workload for packaging is represented by the number of orders, denoted as \( NO \), as it takes a fixed amount of time to package each order. Additionally, assuming that the work speed of all employees is consistent, the work speed in this context and hourly wage corresponding to each of the three operations are represented as \( v_{DT} \), \( v_{NU} \), \( v_{NO} \), and \( c_{DT} \), \( c_{NU} \), \( c_{NO} \) respectively. All the involved parameters and the corresponding operation types are listed in Table 1. We give the total cost as follows.

\[
\text{Cost} = \frac{DT}{v_{DT}} \cdot c_{DT} + \frac{NU}{v_{NU}} \cdot c_{NU} + \frac{NO}{v_{NO}} \cdot c_{NO} \quad (1)
\]

If the orders are to be assigned to \(|J|\) batches, the total horizontal travel distance, denoted as \( DH \), is equal to the batch number, i.e. \(|J|\), multiplied by the round-trip distance from the starting point to the packing point, which is a constant. Similarly, the packaging workload is also a constant. Here we represent the sum of their cost as \( C_0 \) and denote the vertical travelling distance as \( DV = DT - DH \). Further, to simplify the optimisation, let \( \tau = \frac{\nu_{NU} \cdot v_{DT}}{\nu_{NU} \cdot v_{NO} + \nu_{NU} \cdot v_{DT}} \). We modify the Equation (1) as

\[
\text{Cost} = \frac{c_{DT} \cdot v_{DU} + c_{NU} \cdot v_{DU} \cdot (\tau \cdot NU + (1 - \tau) \cdot DV)}{v_{DU} \cdot v_{NU}} + C_0 \quad (2)
\]

Therefore, we only need to optimise \( \tau \cdot NU + (1 - \tau) \cdot DV \). We will formulate it in the modelling part. However, for different numbers of batches, the constant \( C_0 \) in Equation (2) is different, to consider the impact of varying the number of batches, we use the expression \( \tau \cdot NU + (1 - \tau) \cdot DT \) in our numerical experiments in Section 6.

The manual pickers use the Return strategy and the S-shape strategy for routeing in practice. We follow the definitions of Gu, Goetschalckx, and McGinnis (2007) and De Koster, Le-Duc, and Roodbergen (2007). In OBOPP, these strategies are characterised as follows (See Figure 1 for illustration). First, the order picker can only move in vertical or horizontal directions. For the Return strategy, the order picker enters an aisle containing one or more items to be picked. He/she walks down the aisle until the final item required is picked, then returns to the bottom line to the next aisle. The S-shape strategy provides a solution in which the order picker traverses through an aisle containing one or more items to be collected. Then, he/she moves horizontally to the next aisle. If an item picker has completed all the items in a batch, he/she will move to the packing area to unload the items.

The OBOPP is NP-hard. The general OBP has been proven to be NP-hard (Gademann and Velde 2005). In an OBOPP, if we set every item to be the same type, the packing time cost becomes constant, and the problem degenerates into a generic OBP. Thus, the OBOPP is at least as hard as OBP and then proven to be NP-hard.

| Table 1. Cost parameters. |
|---------------------------|
| Workload | Work speed | Hourly wage | Operation type |
| --- | --- | --- | --- |
| \( DT \) | \( v_{DT} \) | \( c_{DT} \) | routeing |
| \( NU \) | \( v_{NU} \) | \( c_{NU} \) | sorting |
| \( NO \) | \( v_{NO} \) | \( c_{NO} \) | packaging |
2.2. Model formulation

For the OBOPP defined above, We give two MIP models corresponding to the Return strategy and the S-shape strategy. The models’ notations are listed below (Table 2):

Variables:

- \( y_{ij} = \begin{cases} 1, & \text{if order } i \text{ is assigned to batch } j \\ 0, & \text{otherwise} \end{cases} \)
- \( z_{jk} = \begin{cases} 1, & \text{if item } k \text{ is in an order that assigned to batch } j \\ 0, & \text{otherwise} \end{cases} \)
- \( \delta_{jb} = \begin{cases} 1, & \text{if there exists item that located on aisle } b \text{ is assigned to batch } j \\ 0, & \text{otherwise} \end{cases} \)

\( d_{jb} = \max\{r_{jb} \mid y_{ij} = 1\} \), which indicates the maximum vertical distance for picking items from batch \( j \) on aisle \( b \).

- \( q_{jb} = \begin{cases} k, & \text{if aisle } b \text{ is the } k\text{th aisle passed in batch } j \\ 0, & \text{otherwise} \end{cases} \)
- \( o_{jb} = \begin{cases} 1, & \text{if aisle } b \text{ passed in batch } j \text{ is numbered by an odd number} \\ 0, & \text{otherwise} \end{cases} \)

\( u_{jb} \): an auxiliary variable to decide the parity of \( q_{jb} \), which satisfies \( q_{jb} = 2u_{jb} + o_{jb} \)

The OBOPP with the Return strategy can be stated as the following MIP:

\[
\begin{align*}
\min & \quad \tau \sum_{j \in J} \sum_{k \in K} z_{jk} + (1 - \tau) \\
& \quad \sum_{j \in J} \sum_{b \in B} 2d_{jb} : (y, z, d) \in P_{ijKB}^R \\
\end{align*}
\]

where polytope \( P_{ijKB}^R \) is given by:

\[
\begin{align*}
\sum_{j \in J} y_{ij} &= 1, \quad \forall i \in I; \\
\sum_{i \in I} y_{ij} &= C, \quad \forall j \in J; \\
M_k z_{jk} &\geq \sum_{i \in I_k} y_{ij}, \quad \forall j \in J, \quad \forall k \in K; \\
d_{jb} &\geq r_{ib} y_{ij}, \quad \forall j \in J, \quad \forall b \in B; \\
\end{align*}
\]

The objective function in (3) aims to minimise the total number of unique items in every batch and the total vertical distance of the item-picker’s picking path. The constraint (4) ensures that each order is assigned to only one batch. The constraint (5) guarantees that each batch does not exceed its capacity limit. The big-M constraint (6) is used to ensure that \( z_{jk} \) indicates whether item \( k \) is included in batch \( j \). Note that the big-M is with a subscript to obtain a more tight model since the sizes of set \( I_k \) may vary. The constraint (7) provides the walking length \( d_{jb} \) for batch \( j \) within aisle \( b \). At last, constraints (8)–(10) define the domain of decision variables.

The OBOPP with the S-shape strategy is stated as follows:

\[
\begin{align*}
\min & \quad \tau \sum_{j \in J} \sum_{k \in K} z_{jk} + (1 - \tau) \\
& \quad \sum_{j \in J} \sum_{b \in B} 2l_{b} o_{jb} : (y, z, \delta, q, o, u) \in P_{ijKB}^S \\
\end{align*}
\]

where polytope \( P_{ijKB}^S \) is given by:

\[
\begin{align*}
\sum_{j \in J} y_{ij} &= 0, \quad \forall i \in I; \\
\sum_{i \in I} y_{ij} &= 0, \quad \forall j \in J; \\
M_b \delta_{jb} &\geq \sum_{i \in I_b} y_{ij}, \quad \forall j \in J, \quad \forall b \in B; \\
\delta_{jb} &\leq \sum_{i \in I_b} y_{ij}, \quad \forall j \in J, \quad \forall b \in B; \\
q_{jb} &\geq \delta_{jb}, \quad \forall j \in J, \quad \forall b \in B; \\
\end{align*}
\]

Table 2. Parameters and indices.

| Parameters and indices: | Description                                                                 |
|------------------------|-----------------------------------------------------------------------------|
| \( I, i \)            | the set of custom orders and its index \( i \in I \);                       |
| \( K, k \)            | the set of items (i.e. the set of items that appears in any of the orders) and its index \( k \in K \); |
| \( B, b \)            | the set of aisles and its index \( b \in B \);                              |
| \( J, j \)            | the set of feasible batches and its index \( j \in J \);                    |
| \( I_k, M_k \)        | the set of orders that include item \( k \) and the cardinality of it, as \( M_k = |I_k| \); |
| \( I_b, M_b \)        | the set of orders that include one or more items located in aisle \( b \) and the cardinality of it, as \( M_b = |I_b| \); |
| \( r_{ib} \)          | the maximum vertical distance of the items on aisle \( b \) contained in order \( i \) from the bottom of the warehouse; |
| \( l_{b} \)           | the length of aisle \( b \);                                               |
| \( C \)               | capacity of the packing point;                                             |
| \( \tau \)            | weight of order packing time-consuming.                                    |
\[ \delta_{jb} q_{jb} = \delta_{jb} \left( \sum_{b'=1}^{b-1} \delta_{jb'} + 1 \right), \quad \forall \, j \in J, \quad \forall \, b \in B; \quad (15) \]
\[ q_{jb} = 2u_{jb} + o_{jb}, \quad \forall \, j \in J, \quad \forall \, b \in B; \quad (16) \]
\[ \delta_{jb} \in \{0, 1\}, \quad \forall \, j \in J, \quad \forall \, b \in B; \quad (17) \]
\[ q_{jb} \in \mathbb{Z}, \quad \forall \, j \in J, \quad \forall \, b \in B; \quad (18) \]
\[ u_{jb} \in \mathbb{Z}, \quad \forall \, j \in J, \quad \forall \, b \in B; \quad (19) \]
\[ o_{jb} \in \{0, 1\}, \quad \forall \, j \in J, \quad \forall \, b \in B. \quad (20) \]

The decision variable \( \delta_{jb} \) in constraint (12) determines whether an aisle \( b \) is passed. Constraints (13) and (14) give upper bounds and lower bounds for variables \( \delta_{jb} \) and \( q_{jb} \), respectively. Non-linear constraint (15) guarantees that all the non-zero ordinal numbers \( q_{jb} \) of a batch \( j \) are an arithmetic sequence with a difference of one. Constraint (16) ensures that \( o_{jb} \) indicates whether or not \( q_{jb} \) is an odd number. Constraint (17)–(20) gives domains of variables \( \delta_{jb}, q_{jb}, u_{jb} \) and \( o_{jb} \).

The models proposed above are overly complex and present a challenge in terms of practical solvability using a solver, especially for larger cases (e.g. more than 100 orders). The inclusion of additional auxiliary variables increases the complexity of these models. Hence, our optimisation strategy is aimed at presenting a simplified easy-to-solve approximate model, which is the driving force behind our statistics-based approach.

3. Statistics-based framework for model approximation

The studies in the literature review Section 1.1.3 investigate the statistical properties of optimisation problems to estimate their optimal values. This has inspired us to adopt a new perspective: considering the objective value of an exact OBOPP model as a random variable. Then our statistical approach is developed in two phases. In Section 3.1, we infer the distribution of the random variables, and in Section 3.2, we propose a correlation-based framework for finding an approximate model with a highly correlated distribution. Moreover, theoretical support is provided to illustrate the effectiveness of this approach.

3.1. Inference by statistics: the distribution of the objective value

To construct a random variable, the sample space must first be determined. Therefore, we denote a solution of the OBOPP that encompasses the allocation information of each order to a specific batch as \( p \). The solution \( p \) can be represented as a \( |J| \)-partitioned permutation as follows:

\[ p = (\{a^1_j\}_{k \in \mathbb{N}_C}, \{a^2_j\}_{k \in \mathbb{N}_C}, \ldots, \{a^k_j\}_{k \in \mathbb{N}_C}), \]

where \( \mathbb{N}_C = \{1, 2, \ldots, C\} \) and \( \{a^k_j\}_{k \in \mathbb{N}_C} \) denotes a set of orders assigned to batch \( j \). The sets of orders satisfy \( \cup_{j \in J} \{a^k_i\}_{k \in \mathbb{N}_C} = I \), \( \{a^k_i\}_{k \in \mathbb{N}_C} \cap \{a^k_j\}_{k \in \mathbb{N}_C} = \emptyset \) for a unequal pair of batches \( i, j \in J \).

**Observation 3.1:** Given the set \( I \), the feasible \( |J| \)-partitioned permutations are in one-to-one correspondence to the feasible integer points of \( P_{IJ} = \{y_{ij} \in \{0, 1\} : \sum_{j \in J} y_{ij} = 1, \sum_{i \in I} y_{ij} = C, i \in I, j \in J \} \). Here \( P_{IJ} \) is the projection of \( P^{R}_{IBJ} \) and \( P^{S}_{IJKB} \) on variable \( y \), i.e. \( \text{Proj}_y(P^{R}_{IBJ}) = \text{Proj}_y(P^{S}_{IJKB}) = P_{IJ} \).

The Observation 3.1 indicates that the solution set \( p \) covers all feasible assignments and has a one-to-one correspondence with the integer points in \( P_{IJ} \). Thus, for a \( |J| \)-partitioned permutation \( \hat{p} \), the corresponding integer point in \( P_{IJ} \) can be represented as \( \hat{y}(\hat{p}) \), and the objective values can be computed by setting \( y \) fixed as \( \hat{y}(\hat{p}) \) in the exact models and solving

\[
\min \left\{ \tau \sum_{j \in J} \sum_{k \in K} z_{jk} + (1 - \tau) \times \sum_{j \in J} \sum_{b \in B} 2d_{jb} : (\hat{y}(\hat{p}), z, d) \in P^{R}_{IBJ} \right\}, \quad (21)\]

and

\[
\min \left\{ \tau \sum_{j \in J} \sum_{k \in K} z_{jk} + (1 - \tau) \times \sum_{j \in J} \sum_{b \in B} l_{b} \delta_{jb} : (\hat{y}(\hat{p}), z, \delta, q, u, o) \in P^{S}_{IJKB} \right\}. \quad (22)\]

corresponding to two routeing strategies. Therefore, the definition of the random variable of the objective value can be provided as follows.

**Definition 3.2:** Let \((\Omega, \mathcal{F}, \mathcal{P})\) be a probability space, where the sample space \( \Omega \) is the set of all feasible \( |J| \)-partitioned permutations, the event space \( \mathcal{F} \) is the power set of \( \Omega \), and \( \mathcal{P}(p) = \frac{1}{2^n} \) for every feasible \( |J| \)-partitioned permutation \( p \). Denote the objective function and polytope as a tuple \( \pi = (f, P) \). Define random variable \( X^\pi(p) \)
Figure 2. Normality of solution distribution under different routing strategies.

as mapping

\[ p \mapsto \min \{ f(y(p), w) : (y(p), w) \in P \}, \]

where \( w \) indicates other variables except for variable \( y \). The polytope \( P \) satisfies \( \text{Proj}_y P = P_I \).

To simplify the notions, we use \( X^R \) and \( X^S \) to denote the random variables of the initial objective functions and polytopes, where \( R \) and \( S \) are the tuples of objective functions and polytopes in (21) and (22). By generating random solution \( p \), we can record the sampled value of \( X^R \) and \( X^S \) and further infer their distribution. Denote \( X^R_i \) and \( X^S_i \) the \( i \)th sampled value. The problem becomes to infer the distribution of \( X^R \) and \( X^S \) in probability space \((\Omega, \mathcal{F}, P)\).

The experiments are organised as a Monte Carlo Sampling study. We randomly generate \( n \) random solutions \( p \) with a real-world data instance (800 orders). The method of sampling from random variable \( X^\pi(p) \) can be expressed as the following three steps:

1. Generate a random permutation of elements in \( I \) and partition the permutation into \(|J|\) feasible batches to maintain \( \hat{p} \). Get the corresponding feasible integer point \( \hat{y}(\hat{p}) \). Go to step 2.
2. Solve the problem \( \min \{ f(\hat{y}(\hat{p}), w) : (\hat{y}(\hat{p}), w) \in P \} \) with \( \hat{y}(\hat{p}) \). Record the objective value in a list.
3. Repeat step 1 until the list has \( n \) records.

The generating process in step 1 ensures the equivalent probability of each solution in the sample space \( \Omega \). We use the Python packages \texttt{ScipyStats} and \texttt{Pingouin} for normality tests. We set \( n = 5000 \) to ensure significant conclusions of the experiments.

The experiment results are shown below. We plot empirical probability density functions (PDF) (See Figure 2) with \( n \) sampled points to show the probability density of \( X^R \) and \( X^S \). To verify the normality, we conduct a Kolmogorov-Smirnov (K-S) test and the results accept the normality hypothesis with \( p \)-values equal to 0.8453 and 0.8236.

3.2. Approximation by pattern: a correlation-based framework

With the definition and inference of the random variables, it becomes possible to formulate model simplification as an optimisation problem. The objective of this problem is to find an approximation model whose corresponding random variable exhibits the highest correlation value with that of the exact models. As a result, by solving the approximate model, high-quality solutions for the exact models can be obtained simultaneously. Meanwhile, it is essential to specify a proper class of approximation models from all the arbitrary ones as the feasible region of this optimisation problem. This class of models must adhere to the basic constraints of order assignments, i.e. the constraints (4) and (5), to cover all feasible solutions, while maintaining a straightforward structure.

Therefore we define this problem as the Maximal Correlation Reformulation problem (MCRP). Denote \( Y^\pi \) with \( \pi = (f_Y, P_Y) \) as the random variable of the objective value of the approximate model. For a given random variable \( X^{\pi_0} \) of an exact model with \( \pi_0 = (f_X, P_X) \), we describe the MCRP as follows.

\[ \max \text{Corr}(Y^\pi, X^{\pi_0}). \]  \hspace{1cm} (23)

Here \( f_Y \) for minimisation is given by:

\[ \sum_{j \in J} \sum_{\eta \in H} \beta_{\eta j} \delta_{j\eta}. \]  \hspace{1cm} (24)
and \( P_Y \) is given by:
\[
\sum_{j \in J} y_{ij} = 1, \quad \forall i \in I; \tag{25}
\]
\[
\sum_{i \in I} y_{ij} = C, \quad \forall j \in J; \tag{26}
\]
\[
M_\eta \delta_{\eta j} \geq \sum_{i \in I_\eta} y_{ij}, \quad \forall j \in J, \quad \forall \eta \in H; \tag{27}
\]
\[
y_{ij} \in \{0, 1\}, \quad \forall i \in I, \quad \forall j \in J; \tag{28}
\]
\[
\delta_{\eta j} \in \{0, 1\}, \quad \forall j \in J, \quad \forall \eta \in H; \tag{29}
\]
with \( \beta_\eta \in R_+, \ M_\eta = |I_\eta| \) and \( I_\eta \in \mathcal{P}(I) \), which is the power set of \( I \) for any \( \eta \in H \). Here \( H \subseteq \{1, 2, 3, \ldots\} \), \( |\mathcal{P}(I)| \) is an index set.

Here \( \text{Corr}(Y^\pi, X^{\pi_0}) \) stands for the correlation coefficient for random variables \( Y^\pi \) and \( X^{\pi_0} \). Constraints (25) and (26) are the original constraints that ensure the uniqueness and the limitation of the assignment. In (24), we construct an objective function as the positive weight sum of \( \delta_{\eta j} \). Further, in (27), with the sets \( I_\eta \) and under the premise of minimising the positive weight sum of \( \delta_{\eta j} \), essentially \( \delta_{\eta j} \) functions as a binary indicator to decide whether the assignment for batch \( j \) contains any element in \( I_\eta \). By solving this minimising problem in (24)–(29), we arrive at the objective value, the physical meaning of which pertains to the weighted sum of the frequency of appearance of the elements of \( I_\eta \) in each batch of an assignment.

By constraints (24)–(29) we specify a class of approximate models. The differences between the approximate models lie in the selection of the coefficients \( \beta_\eta \) and sets \( I_\eta \). In this context, the objective function (24) and constraints (27) assume pivotal roles in enabling the models to approximate. Consequently, the coefficients \( \beta_\eta \) and sets \( I_\eta \) in these equations function as variables in the MCRP, determining the structure of the approximate models within this class, the resulting distribution of the corresponding random variables, and ultimately, the correlation coefficient. Therefore, to streamline representation and enable further investigation, we include them in the concept of \textit{pattern} as follows.

**Definition 3.3:** For a given set \( I \), containing orders of acknowledged contents, a \textit{pattern} of an OBOPP can be represented as a set of pairs comprising coefficients and subsets of \( I \), denoted by \( \{ (\beta_\eta, I_\eta) \}_{\eta \in H} \), and is generated in accordance with specific rules.

Consequently, a well-defined \textit{pattern} directly corresponds to an approximate model, which serves as a solution to the optimisation problem. An instance of a \textit{pattern} and its resulting objective value is provided as follows.

**Example 3.4:** Suppose we have four orders as \( I = \{1, 2, 3, 4\} \), and they need to be divided evenly into two batches. We define the \textit{pattern} with two elements, namely \((1, \{1\})\) and \((1, \{1, 2\})\). Assuming that we obtained a random solution \( p = ((1, 3), (2, 4)) \) through random sampling. With the formulated elements of \( \pi \), it can be computed that \( Y^\pi(p) = 3 \) as \( \delta_{11} = 1, \delta_{12} = 1, \delta_{21} = 1, \) and \( \delta_{22} = 0 \).

For a given set \( I \), containing orders of acknowledged contents, assuming that we have obtained all information regarding all solutions to this problem, we can further derive the value of the indicator \( \delta_{\eta j} \) for each subset \( I_\eta \) of \( I \) and each batch \( j \). In this scenario, the MCRP for this single instance can be regarded as a correlation-based feature selection problem to determine if the subset \( I_\eta \) and its weight should be included in a \textit{pattern} to obtain maximal correlation value. Feature selection is very hard and the mainly used algorithms are heuristics (Li et al. 2017). Furthermore, since we do not know the information of all solutions and there is not only a single instance, giving an exact result or bound of the solution of MCRP is out of the scope of this paper.

Nonetheless, the efficacy of this framework can still be measured through statistical analysis. With the tested normality of \( X^R \) and \( X^S \) in Section 3.1 and tools of multivariate statistical analysis (Härdle and Simar 2007), we present the conditional expectation and upper bound of \( X^{\pi_0} \) in the following propositions.

**Proposition 3.5:** Random variables \( Y^\pi \) and \( X^{\pi_0} \) are defined by \( p \mapsto \min\{f(y(p), w) : (y(p), w) \in P\} \). If \( (X^{\pi_0}, Y^\pi) \) follow bivariate normal distribution
\[
(X^{\pi_0}, Y^\pi) \sim N \left( (\mu_X, \mu_Y), \begin{bmatrix}
\sigma_X^2 & \rho \sigma_X \sigma_Y \\
\rho \sigma_X \sigma_Y & \sigma_Y^2
\end{bmatrix} \right). \tag{30}
\]
Given \( Y^\pi = Y_0 \), the conditional expectation of \( X^{\pi_0} \) is
\[
\mathbb{E}[X^{\pi_0} \mid Y^\pi = Y_0] = \mu_X + \sigma_X \rho \frac{Y_0 - \mu_Y}{\sigma_Y}. \tag{31}
\]

The proof can be driven directly from the definition of bivariate normal distribution random variables.

This proposition clearly indicates that a higher correlation coefficient \( \rho \) signifies a superior solution, given that the provided \( Y_0 \) would be less than \( \mu_Y \), and a higher \( \rho \) would lead to a lower expected value of \( X^{\pi_0} \). For the same purpose, we also provide the confidence interval of \( X^{\pi_0} \) given \( Y^\pi = Y_0 \). For interval \( (-\infty, \xi) \), \( X^{\pi_0} \) has a probability of \( 1 - \alpha \) to be less than \( \xi \).

**Proposition 3.6:** Random variables \( Y^\pi \) and \( X^{\pi_0} \) are defined by \( p \mapsto \min\{f(y(p), w) : (y(p), w) \in P\} \). If \( (X^{\pi_0}, \)
$Y^\pi$) follow bivariate normal distribution (30). Given $Y^\pi = Y_0$, the upper bound $\xi$ that $P(X^{\pi_0} < \xi \mid Y^\pi = Y_0) = 1 - \alpha$ is given by

$$\xi = \sigma_{X|Y} \Phi^{-1}(1 - \alpha) + \mu_{X|Y},$$

where $\Phi(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\xi} e^{-t^2/2} dt$ and

$$\mu_{X|Y} = \mu_X + \sigma_X \rho \frac{Y_0 - \mu_Y}{\sigma_Y},$$

$$\sigma_{X|Y} = \sigma_X \rho \frac{\rho - \sigma_Y}{\sigma_Y}.$$ 

**Proof:** By the assumption of bivariate normal distribution, since

$$f_{X|Y}(X^{\pi_0}) = \frac{f_{Y,X}(Y^\pi, X^{\pi_0})}{f_Y(Y^\pi)},$$

where

$$f_{Y,X}(Y^\pi, X^{\pi_0}) = \frac{1}{2\pi \sigma_Y \sigma_X (1 - \rho)} \exp \left( -\frac{1}{2 (1 - \rho^2)} \left( \frac{(Y^\pi - \mu_Y)^2 \sigma_X^2 + (X^{\pi_0} - \mu_X)^2 \sigma_Y^2 - 2 \rho (Y^\pi - \mu_Y) (X^{\pi_0} - \mu_X) \sigma_Y \sigma_X)}{\sigma_X^2} \right) \right),$$

and

$$f_Y(Y^\pi) = \frac{1}{\sqrt{2\pi \sigma_Y^2}} \exp \left( -\frac{(Y^\pi - \mu_Y)^2}{2 \sigma_Y^2} \right).$$

The conditional distribution follows

$$X^{\pi_0} \mid Y^\pi = Y_0 \sim N \left( \mu_X + \rho \frac{\sigma_Y}{\sigma_X} (Y_0 - \mu_Y), (1 - \rho^2) \frac{\sigma_X^2}{\sigma_Y^2} \right).$$

The equation

$$P(X^{\pi_0} < \xi \mid Y^\pi = Y_0) = 1 - \alpha$$

lead to

$$\Phi \left( \frac{\xi - \mu_{X|Y}}{\sigma_{X|Y}} \right) = 1 - \alpha.$$

Then, with the known upper critical value we have the result

$$\xi = \sigma_{X|Y} \Phi^{-1}(1 - \alpha) + \mu_{X|Y}.$$

If $(X^\pi, Y^\pi)$ does not follow a bivariate normal distribution or any known distribution, we can still estimate the upper bound of the confidence interval of $X^{\pi_0}$ on condition that $Y^\pi = Y_0$ by the empirical distribution function.

**Proposition 3.7:** Random variables $Y^\pi$ and $X^{\pi_0}$ are defined by $p \mapsto \min_f(y(p), w) : (y(p), w) \in P$. Let $((X_1, Y_1), \ldots, (X_n, Y_n))$ be independent, identically distributed random variables with the common cumulative distribution function of $F(u, v)$. Then the empirical distribution function is defined as $\hat{F}_n(u, v) = \frac{1}{n} \sum_{i=1}^n 1_{(X_i \leq u, Y_i \leq v)}$, where $1_A$ is the indicator of event $A$. Given $Y^\pi \leq Y_0$ and a real number $\xi$,

$$P(X^{\pi_0} \leq \xi \mid Y^\pi \leq Y_0) = \frac{\hat{F}_n(\xi, Y_0)}{\hat{F}_n(Y_0)} = \frac{\sum_{i=1}^n 1_{(X_i \leq \xi, Y_i \leq Y_0)}}{\sum_{i=1}^n 1_{Y_i \leq Y_0}}.$$

The proof can be driven directly from the definition of the cumulative distribution function.

### 4. Rule-based pattern generation: construct solutions for the MCRP

A correlation-based framework for approximate models is proposed in the previous section. In this section, we present a solution approach to this framework, utilising a rule-based methodology to identify approximate models. In Section 4.1, we illustrate the design of this approach. In Section 4.2, we present three resultant rules and examine them for selection.

#### 4.1. The design of generation rules

One reason for the success of past heuristics in the OBP is that they capture important features of the problem, such as order size and location. The heuristics cluster orders by these features. For example, the Seed heuristic (De Koster, Van der Poort, and Wolters 1999) conducts the saving rules by putting similar orders together (i.e., orders on the same picking route, or sharing the same aisle). Due to the fact that similar orders are more likely to be picked without increment of the travelling distance, high-quality solutions can then be found.

The approximate model can perform similar rules better with a properly generated pattern. In this context, the pattern is assigned with the physics meaning of a weighted cluster of order sets sharing a specific similarity. Therefore, by minimising the weighted sum of the frequency of appearance of $I_0$, a feasible order assignment of the most concentrated similarity would be found. Finally, by examining the correlation coefficient value
with the exact models, the strength of certain features for measuring similarity is evaluated for selection.

### 4.2. Generation rules and selection

In the following (See Table 3), we provide three kinds of rules and the corresponding correlation coefficient values (refer as R and S for Return and S-shape strategy) computed in the Monte Carlo experiment with \( n = 5000 \).

The scatter plots (refer to Figures 3 and 4) are shown below. The correlation coefficient value increased from 0.23 to 0.72 for the Return strategy and from 0.16 to 0.97 for the S-shape strategy, and the random variables still follow bivariate normal distributions. These distinct values reflect the varying effectiveness of the second step in each rule, which utilise different features to generate patterns. With the results above, we can select rule AP3 as the optimal one. However, we will conduct experiments to prove our selection.

### 5. Integrated heuristic

To further improve the order assignment given by the approximate model described and selected in the previous section, we consider a variant of local search based on the empirical results of Section 4.2, as an integrated heuristic. In this section, the design of this heuristic is

| Index | Generation rule | R   | S   |
|-------|-----------------|-----|-----|
| AP1   | Step 1. Let \( H \) be a set of items that are contained by at least one order \( i \in I \). Step 2. Let \( \beta = 1 \) and \( l = \{ i \mid \text{order } i \text{ contains } \eta \} \), \( \forall \eta \in H \). Step 3. Let \( \left( \beta, l \right) \) be the pattern given known \( l \). Generate the objective function (24) and constraints (27) for model corresponding to \( Y^p \). | 0.23 | 0.16 |
| AP2   | Step 1. Let \( H \) be a set of aisles that store at least one item requested by orders in \( I \). Step 2. Let \( \beta = 1 \) and \( l = \{ i \mid \text{order } i \text{ contains } \eta \} \), \( \forall \eta \in H \). Step 3. The same as Step 3 in AP1. | 0.71 | 0.84 |
| AP3   | Step 1. Let \( H \) be a set of aisles that store at least one item requested by orders in \( I \). Step 2. Let \( \beta = \lambda \) and \( l = \{ i \mid \text{order } i \text{ contains } \eta \} \), \( \forall \eta \in H \). Step 3. The same as Step 3 in AP1. | 0.72 | 0.97 |

**Figure 3.** Solution distribution using Return strategy.

(a) AP1 rule with Corr=0.23.  
(b) AP2 rule with Corr=0.71.  
(c) AP3 rule with Corr=0.72.

**Figure 4.** Solution distribution using S-shape strategy.

(a) AP1 rule with Corr=0.16.  
(b) AP2 rule with Corr=0.84.  
(c) AP3 rule with Corr=0.97.
first illustrated, followed by a detailed presentation of its implementation procedure.

A well-defined neighbourhood is vital to designing a local search-type algorithm for the OBOPP. The OBOPP is high-dimensional, and the search space is vast. Therefore, if we naively define the neighbourhood (e.g., arbitrarily swap two orders from separate batches), we might cause an inefficient searching procedure with a significant increase in computation time.

The well-defined neighbourhood can be determined based on the empirical results of the examination in Section 4.2. The examination has verified a close relationship between the number of visited aisles in an order assignment and the objective value. It indicates that we can select orders with similar aisle distributions to avoid increasing the number of visited aisles but lowering the real objective value by swapping them. Therefore, we consider \( h \)-different pairs. Denote \( A_1 \) and \( A_2 \) the set of aisles visited in orders \( i_1 \) and \( i_2 \). A pair of order \( (i_1, i_2) \) is called \( h \)-different pairs if \(|A_1 \setminus A_2| \leq h \) and \(|A_2 \setminus A_1| \leq h \). Finally, we also set the minimum order size \( m \) to ensure that we only generate the order pairs that have the potential to cause a significant decrease in objective value.

We describe the algorithm as the valuable pairs generation (VPG) method:

\begin{algorithm}
\caption{Valuable pairs generation}
\begin{algorithmic}
\Require order sequence \( i_1, i_2, \ldots, i_n \), empty pair list \( T \)
\For {\( p = 1 : n - 1 \)}
\For {\( q = p + 1 : n \)}
\If {\( |A_p \setminus A_q| \leq h \), \( |A_q \setminus A_p| \leq h \) and orders \( i_p, i_q \) have more than \( m \) items} \Then
\State Add \((i_p, i_q)\) to pair list \( T \)
\EndIf
\EndFor
\EndFor
\end{algorithmic}
\end{algorithm}

We apply the pair list to the solution maintained by an approximate model. In this process, we scan the list and swap all pairs in separate batches which can decrease the objective value. We set \( h = 2 \) and \( m = 1 \) to achieve a satisfactory trade-off between the full strength and efficiency of this algorithm in practice.

6. Experiment results

6.1. Experiment design

In numerical experiments, we implement our methods, including AP1, AP2, and AP3, which stand for the approximation models proposed in Section 4, and the exact method, which stands for solving formulation (3) and (11) with a MIP solver, and AP3+VPG, which denotes the VPG algorithm with AP3 providing an initial solution. The notations and parameters for the experiments are listed in Table 4.

For comparison, since our problem is newly raised, we examine the mainstream and frequently-used heuristics of OB on our problem. These heuristics include the seed algorithms (Seed), an efficient class of algorithms with many variants (De Koster, Van der Poort, and Wolters 1999; Ho, Su, and Shi 2008; Ho and Tseng 2006), of which we choose the most efficient one, the iterated local search (ILS) algorithm, which is also an important heuristic for OB (Henn et al. 2010; Van Gils et al. 2019), and the CWI and CWII algorithm (De Koster, Van der Poort, and Wolters 1999), which are commonly used as benchmarks in comparison experiments (Hong and Kim 2017; Matusiak, De Koster, and Sari 2017; Žulj and Kramer and Schneider 2018). To guarantee that ILS function to its best potential, parameter adjustment is done. In addition, we include picking-only exact and heuristic methods for both strategies, which stand for removing all the \( z_{jk} \) variables and their associated constraints from the exact models and considering only the travelling distance in the objective functions in (3) and (11). The heuristics are implemented not only through the original picking-only approach but also through our integrated approach, in order to evaluate their comparative effectiveness. In the notation of experimental results, we distinguish these two approaches by adding the suffix ‘-PO’ to represent picking-only methods. The meaning of conducting experiments for both picking-only and integrated problems is to examine the benefits of considering order picking and packing, and a statistical-based framework through comparisons. The picking-only problems may produce solutions to be utilised for further calculating the packing costs, which plus the picking-only costs may act as a baseline to evaluate the results of jointly considering picking and packing problems, and of the integrated problems within a statistical-based framework.

The datasets were acquired from a warehouse in China. The datasets contain information about the warehouse’s layout. The warehouse contains 541 storage locations for 6413 unique items. A storage location’s length is one length unit (LU). Aisles range in length from 1 to 36 LU and are placed in decreasing order from left to right. All the aisles align along the bottom line of the warehouse. The horizontal travel distance of the warehouse is 48 LU. The depot is located on the left side of the aisles, while the packing point is located on the right side of the aisles. In practice, the pickers and packers are paid equally, i.e. \( c_{DT} = c_{NU} \). The parameter \( \tau \) is then
Table 4. Experiments notations and parameters.

| Profiles             | Notations and parameters |
|----------------------|---------------------------|
| Algorithms           | Exact, AP1, AP2, AP3, AP3+VPG, ILS, Seed, CWI, CWII, Exact-PO, ILS-PO, Seed-PO, CWI-PO, CWII-PO |
| Routing policy       | Return strategy, S-shape strategy |
| The number of orders (N) | 480, 600, 720, Test(800), 840, 960, 1080, 1200 |
| Packing capacity (C)  | 10, 15, Test(16), 20 |

determined by the work speeds and set to a value of 0.4. Since the manual work speeds do not vary over time in the same environment, we only consider this fixed value.

There are 800 orders in both datasets 1 and 2. As mentioned in the previous sections, we use dataset 1 as a statistical object. In this experiment, we use dataset 2 as real-world test data (Test(800)) to examine our method. Each order has an average of 2.8 items in it. The number of items to unique items ratio is approximately 3:2. All other test data are randomly generated with the same parameters, including the item location, the average of items in an order, and the ratio of the number of items to unique items. First, select a set of unique items arbitrarily from all item types. Assuming that all of the items in the order are empty slots, divide them by n numbers from the quantity of the slot, where n is the number of orders. Finally, randomly place the unique items in the slots.

We use the Python 3.8 language and the academic version of GUROBI9.5.1 for MIP solving. The programs run on a PC (Intel i7 2.60 gigahertz CPU, 16-gigabyte memory) with Windows 10.0. The time limit for solving all of the exact models is set to 1000 seconds. The time limit to solve our approximate model is set to 100 seconds in comparisons within the framework and 200 seconds in comparisons with other heuristics. Each experiment is formed of 20 random instances.

To evaluate experiment results, we denote Obj, LB, and Gap as the objective value, the linear relaxation value, and the integrality gap of the exact models. Let CPU be the running time of algorithms. Finally, we use the Obj ratio to compare the solution quality of two algorithms, which is the ratio of their objective values.

6.2. Experiment results

6.2.1. Comparison within the framework

We first compare our simplified model with the exact models and other approximate models proposed in this paper to examine its improvement. Table 5 shows the results. It indicates that under the Return strategy, AP3 yields a reduction in the objective value of 27–49% (with an average of 33%) and 25–42% (with an average of 33%) in comparison to the Exact method and AP1, respectively. Additionally, when compared to AP2, AP3 demonstrates an average improvement of 2%, which aligns with the minimal difference in their correlation coefficients. Under the S-shape strategy, AP3 results in a substantial reduction in objective value compared to the Exact method and AP1, with a range of 73–93% (with an average of 83%) and 46–69% (with an average of 57%), respectively. Furthermore, compared to the AP2 method, AP3 results in a significant average decrease of more than 5%, which corresponds to the improvement in their correlation coefficients from 0.84 to 0.97.

The significant improvement in solving efficiency between the exact and the approximate models is due to the model simplification done within the framework since it is their only difference. Thus the effectiveness of this framework can be validated. Additionally, the performance of the approximate models AP1, AP2, and AP3 is consistently aligned with their respective correlation coefficients in Table 3. This outcome provides empirical evidence for the theoretical Proposition 3.5 that a higher correlation coefficient indicates a superior solution.

6.2.2. Comparison with exact method on small instances

To illustrate the approximation accuracy of our model with the exact ones and the distance between them, we present a comprehensible experiment on small instances (See Table 6). The limitation on the size of instances is because the solving time of exact modes would grow rapidly to thousands of seconds if the instance were slightly larger. We utilise the GUROBI solver to compare the instance solved to optimal with the AP3 approximation model. It can be observed that the results of the approximation method maintain a shorter error distance from the exact solution under two routeing strategies.

6.2.3. Comparison with four commonly used picking-only algorithms

Tables 7 and 8 show the test results under Return and S-shape routeing strategies. The Obj ratios are all comparisons between AP3+VPG and other algorithms. In the comparison of solution quality in both tables, the AP3+VPG retained overall supremacy, which is about 5% and 10% ahead of the second place on average, respectively. The CWII-PO remains the second-best heuristic in the majority of scenarios, implying it is a highly powerful heuristic. This outcome is consistent with previous experiments published in the literature. The ILS-PO is the
Table 5. Comparison within the framework.

| C   | N   | Exact | AP1 | AP2 | AP3 | Obj ratio Exact | AP1 | AP2 | AP3 | Obj ratio |
|-----|-----|-------|-----|-----|-----|-----------------|-----|-----|-----|----------|
|     |     |       |     |     |     |                 |     |     |     |          |
|     |     |        |     |     |     |                 |     |     |     |          |
| 10  | 480 | 8636.92 | 0.74 | 2279.23 | 8369.48 | 6677.32 | 1.29 | 1.25 | 1.02 | 9499.12 | 0.83 | 1569.33 | 8057.00 | 5773.56 | 5505.40 | 1.73 | 1.46 | 1.05 |
|     |     | 600   | 11,438.8 | 0.76 | 2702.69 | 10,563.80 | 8413.32 | 1.38 | 1.28 | 1.02 | 12,159.76 | 0.83 | 2074.28 | 10,262.60 | 7099.36 | 6854.76 | 1.77 | 1.50 | 1.04 |
|     |     | 720   | 12,953.56 | 0.76 | 3128.53 | 12,905.96 | 10,151.72 | 9964.08 | 1.30 | 1.30 | 1.02 | 14,688.40 | 0.83 | 2402.25 | 12,431.60 | 8548.08 | 8224.32 | 1.79 | 1.51 | 1.04 |
|     |     | 840   | 15,101.92 | 0.77 | 3544.65 | 15,005.52 | 11,741.28 | 10,432.80 | 1.32 | 1.32 | 1.02 | 16,959.92 | 0.83 | 2807.04 | 14,389.08 | 9933.92 | 9454.88 | 1.80 | 1.52 | 1.05 |
|     |     | 960   | 17,163.76 | 0.77 | 3962.71 | 17,221.56 | 13,487.72 | 13,003.52 | 1.32 | 1.32 | 1.04 | 19,361.12 | 0.83 | 3319.48 | 12,431.60 | 8548.08 | 8224.32 | 1.83 | 1.58 | 1.06 |
|     |     | 1080  | 21,095.6 | 0.79 | 4379.22 | 19,689.00 | 15,164.44 | 14,587.72 | 1.32 | 1.35 | 1.04 | 22,103.04 | 0.83 | 3737.93 | 19,100.88 | 12,808.64 | 12,083.32 | 1.82 | 1.59 | 1.05 |
| 15  | 480 | 7212.2 | 0.75 | 1818.13 | 7140.68 | 5774.28 | 5596.12 | 1.29 | 1.28 | 1.03 | 7557.68 | 0.84 | 1200.70 | 6696.44 | 4580.80 | 4406.24 | 1.72 | 1.52 | 1.04 |
|     |     | 600   | 9021.04 | 0.76 | 2120.47 | 8984.68 | 7053.12 | 6982.80 | 1.29 | 1.29 | 1.01 | 10,123.84 | 0.85 | 1504.70 | 8329.12 | 5676.20 | 5491.96 | 1.84 | 1.52 | 1.03 |
|     |     | 720   | 10,757.32 | 0.77 | 2433.95 | 10,936.20 | 8424.12 | 8302.88 | 1.30 | 1.32 | 1.01 | 12,134.16 | 0.85 | 1799.63 | 10,214.64 | 6739.76 | 6542.68 | 1.85 | 1.56 | 1.03 |
|     |     | 840   | 12,229.36 | 0.78 | 2739.07 | 12,932.52 | 9921.32 | 9599.32 | 1.27 | 1.35 | 1.03 | 14,165.92 | 0.85 | 2102.98 | 12,031.56 | 7999.84 | 7675.08 | 1.85 | 1.57 | 1.04 |
|     |     | 960   | 14,223.64 | 0.79 | 3037.68 | 14,821.00 | 11,360.80 | 10,989.12 | 1.32 | 1.35 | 1.03 | 16,261.60 | 0.85 | 2399.62 | 13,803.04 | 9083.64 | 8638.24 | 1.88 | 1.60 | 1.05 |
|     |     | 1080  | 15,556.12 | 0.79 | 3430.68 | 16,492.72 | 12,681.36 | 12,465.36 | 1.31 | 1.36 | 1.02 | 18,300.32 | 0.86 | 2702.50 | 15,802.72 | 10,319.08 | 9889.60 | 1.88 | 1.60 | 1.04 |
|     |     | 1200  | 17,335.36 | 0.79 | 3641.70 | 18,993.48 | 14,053.24 | 13,677.80 | 1.27 | 1.39 | 1.03 | 20,238.48 | 0.86 | 3000.60 | 17,568.72 | 11,405.16 | 10,749.24 | 1.88 | 1.63 | 1.06 |
| 16  | 480 | 12,645.60 | 0.80 | 2533.60 | 11,877.6 | 8993.59 | 9028.40 | 1.40 | 1.32 | 1.00 | 12,853.60 | 0.85 | 1718.52 | 11,191.2 | 6982.20 | 6628.00 | 1.94 | 1.69 | 1.05 |
|     |     | 600   | 16,250.36 | 0.81 | 3064.18 | 16,824.72 | 12,196.04 | 11,886.68 | 1.37 | 1.42 | 1.03 | 17,303.84 | 0.86 | 2425.98 | 15,149.40 | 9604.32 | 8981.64 | 1.93 | 1.69 | 1.07 |
third-best in most cases and slightly better than CWII-PO in the scenarios of small order numbers under the Return strategy. We plot line charts to visualise the domination AP3+VPG on other algorithms and the varying performance of heuristics under different order sizes (See Figure 5).

Taking computing time into consideration, the AP3+VPG grows slowly with the number of orders, from 200 seconds to around 225 seconds, implying that it is practical in real operation processes. This time cost is a good result for integrated branch-and-bound algorithms. The ILS-PO is the fastest algorithm, but this does not compensate for its solution quality in large-scale scenarios. The time cost of the CWII-PO exceeds that of AP3+VPG when the problem size exceeds 800. It also shows a quadratic growth of computing time as the problem scale increases. This result is consistent with the description in De Koster, Van der Poort, and Wolters (1999).

Moreover, we can also find the improvement made by the VPG heuristic, looking back to Table 5. This also shows the effectiveness of our statistical analyses. Furthermore, the results also reveal that as the capacity limit \( C \) increases, the AP3+VPG algorithm demonstrates superior performance for a fixed number of orders. This could be beneficial if the warehouse system improves its efficiency by increasing the capacity limit. Finally, all algorithms perform better under the S-shape strategy, which means we can employ the S-shape strategy for routeing in the real application of this problem.

6.2.4. Comparison with four commonly used algorithms extended to the integrated problem

The two Tables 9 and 10 show the performance of the heuristic algorithms when they are implemented in our integrated problem. The ranking and running time of the algorithms did not notably change. The performance of ILS and Seed did not vary significantly, while CWI and CWII deteriorated slightly. AP3+VPG remained the best algorithm, with an average improvement of 7% and 16% over the second-best algorithm.

Upon analysing the data presented in Tables 7 and 8 with that in Tables 9 and 10, it is observed that the best performance of the comparison algorithms of the picking-only approach slightly outperforms that of the integrated approach. Specifically, only using the integrated approach cannot ensure a better output. This phenomenon can be attributed to the experimental design of the picking-only approach, wherein the \( z_{jk} \) variables are removed. Although this may not produce a globally optimal solution, the resulting models benefit from simplicity, which is further augmented by traditional heuristic algorithms. Conversely, the models of the integrated approach suffer from over-complexity, which serves as the motivation behind our statistical method.

Figure 5. Average cost per order comparison of Return routeing strategy (left) and S-shape routeing strategy (right).
Table 7. Return strategy in picking-only comparison.

| C   | N   | AP3+i-VPG+ | ILS-PO | Seed-PO | CWI-PO | CWII-PO | Exact-PO |
|-----|-----|------------|--------|---------|--------|---------|----------|
|     |     | AP3+i-VPG+ | ILS-PO | Seed-PO | CWI-PO | CWII-PO | Exact-PO |
|     |     | Obj       | CPU    | Obj     | CPU    | Obj ratio | Obj     | CPU    | Obj     | CPU    | Obj     | CPU    | Obj ratio | Obj     | CPU    | Obj     | CPU    | Obj ratio |
| 10  | 480 | 6444.16   | 202.40 | 6777.80 | 9.94   | 1.05     | 9199.60 | 12.96  | 1.43     | 7643.28 | 16.83  | 1.19     | 6868.08 | 41.41   | 1.07     | 9072.31 | 1000.00 | 1.41     |
| 600 | 7992.76 | 204.11 | 8511.92 | 9.95 | 1.06 | 11282.76 | 19.47 | 1.41 | 9419.24 | 27.61 | 1.18 | 8371.64 | 66.17 | 1.05 | 11087.49 | 1000.00 | 1.39     |
| 720 | 9539.92 | 206.43 | 10253.24 | 10.37 | 1.07 | 13465.56 | 26.63 | 1.41 | 11154.64 | 39.64 | 1.17 | 9962.76 | 97.40 | 1.04 | 13246.24 | 1000.00 | 1.39     |
| 840 | 10950.92 | 209.02 | 12006.40 | 11.06 | 1.10 | 15613.16 | 38.15 | 1.43 | 12899.04 | 63.75 | 1.18 | 11350.00 | 161.02 | 1.04 | 14799.67 | 1000.00 | 1.35     |
| 960 | 12416.72 | 211.81 | 13796.84 | 11.42 | 1.11 | 17740.44 | 47.74 | 1.43 | 14717.48 | 83.63 | 1.19 | 12942.60 | 207.89 | 1.04 | 16613.80 | 1000.00 | 1.34     |
| 1080 | 13779.68 | 215.04 | 15499.40 | 13.26 | 1.12 | 19836.24 | 61.54 | 1.44 | 16437.88 | 111.58 | 1.19 | 14360.04 | 299.67 | 1.04 | 19902.87 | 1000.00 | 1.44     |
| 1200 | 15275.80 | 219.80 | 17221.12 | 14.37 | 1.13 | 21980.16 | 76.32 | 1.44 | 18191.01 | 161.60 | 1.19 | 15831.72 | 401.29 | 1.04 | 22292.29 | 1000.00 | 1.46     |

Note: * The column AP3+i-VPG is sourced directly from the integrated approach results in Table 9 for comparison.
| C  | N    | AP3 | CPU | Obj  | AP3 | CPU | Obj  | Seed-PO | CPU | Obj  | CWI-PO | CPU | Obj  | CWII-PO | CPU | Obj  | Exact-PO | CPU | Obj  |
|----|------|-----|-----|------|-----|-----|------|---------|-----|------|---------|-----|------|----------|-----|------|-----------|-----|------|
| 10 | 480  | 5466.08 | 201.72 | 6536.12 | 4.74 | 1.20 | 8542.80 | 3.68 | 6831.16 | 7.27 | 1.25 | 6140.72 | 25.96 | 1.09 | 7344.72 | 10000.00 | 1.34 |
|    | 5249.08 | 202.71 | 8243.68 | 4.74 | 1.21 | 10537.36 | 6.23 | 1.55 | 9620.36 | 19.37 | 1.25 | 7940.40 | 62.50 | 1.10 | 9109.68 | 10000.00 | 1.34 |
| 720 | 8122.40 | 204.05 | 9913.28 | 4.91 | 1.22 | 16688.40 | 8.37 | 1.56 | 11586.24 | 37.10 | 1.55 | 10169.12 | 125.35 | 1.09 | 12814.88 | 10000.00 | 1.38 |
| 840 | 9294.16 | 205.89 | 14613.44 | 11.30 | 1.55 | 11475.56 | 205.27 | 1.23 | 11485.76 | 10000.00 | 1.45 |
| 1080 | 11844.72 | 211.34 | 18542.20 | 19.91 | 1.26 | 14685.52 | 94.80 | 1.24 | 17233.04 | 10000.00 | 1.45 |
| 1200 | 13129.44 | 214.95 | 20657.88 | 23.81 | 1.28 | 16177.80 | 119.62 | 1.23 | 17239.92 | 10000.00 | 1.45 |
| 15 | 480  | 4386.08 | 202.11 | 5382.56 | 6.08 | 1.23 | 6901.32 | 4.73 | 5697.60 | 8.50 | 1.30 | 4913.60 | 34.36 | 1.12 | 5140.16 | 10000.00 | 1.17 |
| 600 | 5448.84 | 203.27 | 6788.80 | 6.06 | 1.25 | 9484.08 | 7.50 | 1.56 | 6962.28 | 18.11 | 1.28 | 6000.88 | 65.30 | 1.10 | 6124.72 | 10000.00 | 1.12 |
| 720 | 6465.56 | 204.96 | 8159.44 | 6.57 | 1.26 | 10680.00 | 10.04 | 1.56 | 8226.48 | 32.61 | 1.27 | 7029.36 | 91.02 | 1.09 | 9358.48 | 10000.00 | 1.45 |
| 840 | 7572.00 | 207.13 | 9631.56 | 6.02 | 1.27 | 11764.68 | 13.62 | 1.55 | 9599.68 | 41.26 | 1.27 | 8129.84 | 136.79 | 1.07 | 10375.28 | 10000.00 | 1.37 |
| 900 | 8508.92 | 210.45 | 11069.16 | 6.79 | 1.30 | 13294.64 | 18.36 | 1.56 | 10825.56 | 58.02 | 1.27 | 9165.88 | 217.39 | 1.08 | 12905.12 | 10000.00 | 1.52 |
| 1080 | 9653.24 | 213.84 | 14246.28 | 6.46 | 1.29 | 15003.60 | 24.20 | 1.55 | 12050.16 | 82.90 | 1.27 | 10294.72 | 333.45 | 1.07 | 14516.65 | 10000.00 | 1.50 |
| 1200 | 10510.24 | 216.40 | 13971.40 | 7.25 | 1.33 | 16529.04 | 28.63 | 1.57 | 13329.48 | 106.02 | 1.27 | 11263.80 | 430.75 | 1.07 | 16205.75 | 10000.00 | 1.54 |
| 16 | Test(800) | 6903.20 | 206.89 | 9002.00 | 5.74 | 1.30 | 10994.00 | 13.48 | 1.59 | 8905.60 | 44.93 | 1.29 | 7538.00 | 125.03 | 1.09 | 10728.00 | 10000.00 | 1.55 |
| 20 | 480  | 3715.32 | 202.34 | 4705.36 | 6.07 | 1.27 | 5820.88 | 5.15 | 1.57 | 4908.16 | 7.53 | 1.32 | 4214.04 | 29.52 | 1.13 | 4972.32 | 10000.00 | 1.34 |
| 600 | 4570.80 | 203.92 | 5987.12 | 7.16 | 1.31 | 7178.08 | 9.06 | 1.57 | 6041.24 | 19.01 | 1.32 | 5152.80 | 79.27 | 1.13 | 5434.56 | 10000.00 | 1.19 |
| 720 | 5427.88 | 206.05 | 7200.68 | 6.56 | 1.33 | 8535.84 | 11.37 | 1.57 | 7171.04 | 20.43 | 1.32 | 6075.88 | 83.35 | 1.12 | 5912.95 | 10000.00 | 1.09 |
| 840 | 6324.72 | 208.35 | 8411.92 | 7.51 | 1.33 | 9907.92 | 15.89 | 1.57 | 8303.16 | 33.82 | 1.31 | 6954.72 | 147.77 | 1.10 | 7692.80 | 10000.00 | 1.22 |
| 900 | 7142.36 | 211.84 | 9714.16 | 7.72 | 1.36 | 11284.40 | 21.17 | 1.58 | 9727.40 | 53.82 | 1.31 | 7890.96 | 231.74 | 1.10 | 8339.05 | 10000.00 | 1.17 |
| 1080 | 8037.04 | 216.31 | 10855.88 | 8.37 | 1.35 | 12564.64 | 26.93 | 1.56 | 10462.32 | 76.48 | 1.30 | 8628.08 | 324.61 | 1.07 | 10186.97 | 10000.00 | 1.27 |
| 1200 | 8812.24 | 219.35 | 12156.48 | 8.07 | 1.38 | 13903.64 | 32.99 | 1.58 | 11557.08 | 100.44 | 1.31 | 9532.12 | 451.88 | 1.08 | 11765.60 | 10000.00 | 1.34 |

Note: * The column AP3 + VPG is sourced directly from the integrated approach results in Table 10 for comparison.
Table 9. Return strategy in integrated comparison.

| C  | N   | Obj | CPU | AP3+VPG | ILS | Obj | CPU | Obj ratio | Seed | Obj | CPU | Obj ratio | CWI | Obj | CPU | Obj ratio | CWII | Obj | CPU | Obj ratio |
|----|-----|-----|-----|---------|-----|-----|-----|-----------|------|-----|-----|-----------|-----|-----|-----|-----------|-----|-----|-----|-----------|
|    |     |     |     |         |     |     |     |           |      |     |     |           |     |     |     |           |     |     |     |           |
| 10 | 480 | 6444.16 | 202.40 | 6700.92 | 8.58 | 1.04 | 9145.40 | 11.79 | 1.42 | 7680.28 | 13.92 | 1.19 | 6944.48 | 33.51 | 1.08 |
| 10 | 600 | 7992.76 | 204.11 | 8421.92 | 10.02 | 1.05 | 11,331.48 | 18.61 | 1.42 | 9662.08 | 24.16 | 1.21 | 8542.72 | 63.64 | 1.07 |
| 10 | 720 | 9539.92 | 206.43 | 10,267.40 | 9.90 | 1.08 | 13,534.68 | 27.10 | 1.42 | 11,543.68 | 39.76 | 1.21 | 10,174.64 | 109.01 | 1.07 |
| 10 | 840 | 10,950.92 | 209.02 | 11,900.00 | 11.30 | 1.09 | 15,515.60 | 37.15 | 1.42 | 13,214.44 | 59.63 | 1.21 | 11,568.00 | 171.11 | 1.06 |
| 10 | 960 | 12,416.72 | 211.81 | 13,812.16 | 12.68 | 1.11 | 17,842.28 | 53.23 | 1.44 | 15,201.36 | 100.29 | 1.21 | 13,169.48 | 304.79 | 1.06 |
| 10 | 1080 | 13,779.68 | 215.04 | 15,518.88 | 13.91 | 1.13 | 21,955.40 | 77.22 | 1.44 | 18,787.44 | 157.98 | 1.23 | 16,189.92 | 483.28 | 1.06 |
| 10 | 1200 | 15,275.80 | 219.80 | 17,421.76 | 12.88 | 1.14 | 23,166.08 | 80.22 | 1.44 | 20,282.96 | 169.08 | 1.23 | 17,570.44 | 511.71 | 1.06 |
| 15 | 480 | 5387.88 | 202.90 | 5591.80 | 10.53 | 1.04 | 7791.12 | 13.00 | 1.45 | 6601.92 | 13.85 | 1.23 | 5873.84 | 37.06 | 1.09 |
| 15 | 600 | 6646.68 | 204.82 | 7040.28 | 10.82 | 1.06 | 9538.36 | 20.48 | 1.44 | 8235.24 | 23.38 | 1.24 | 7166.40 | 69.75 | 1.08 |
| 15 | 720 | 7906.00 | 207.39 | 8468.24 | 13.13 | 1.07 | 11,375.08 | 29.95 | 1.44 | 9797.04 | 37.77 | 1.24 | 8547.64 | 117.41 | 1.08 |
| 15 | 840 | 9130.92 | 209.80 | 10,047.08 | 12.51 | 1.10 | 13,161.88 | 41.10 | 1.44 | 11,419.72 | 56.45 | 1.25 | 9860.36 | 181.61 | 1.08 |
| 15 | 960 | 10,314.32 | 213.53 | 11,458.64 | 15.26 | 1.11 | 14,933.88 | 57.88 | 1.45 | 13,059.36 | 88.23 | 1.27 | 11,080.60 | 302.79 | 1.07 |
| 15 | 1080 | 11,530.40 | 217.55 | 12,960.84 | 16.51 | 1.12 | 16,827.40 | 72.19 | 1.46 | 14,689.92 | 121.85 | 1.27 | 12,457.32 | 420.92 | 1.08 |
| 15 | 1200 | 12,711.28 | 222.78 | 14,459.36 | 15.71 | 1.14 | 18,434.52 | 84.63 | 1.45 | 16,156.44 | 145.88 | 1.27 | 13,589.12 | 511.71 | 1.07 |
| 16 | Test(800) | 8505.60 | 209.25 | 9358.40 | 16.10 | 1.10 | 12,654.40 | 41.48 | 1.49 | 10,832.40 | 52.95 | 1.27 | 9305.20 | 170.48 | 1.09 |
| 20 | 480 | 4671.08 | 202.40 | 4905.60 | 10.73 | 1.05 | 6744.52 | 13.91 | 1.44 | 5858.08 | 13.74 | 1.25 | 5188.60 | 39.50 | 1.11 |
| 20 | 600 | 5782.04 | 205.60 | 6185.20 | 12.93 | 1.07 | 8422.40 | 23.35 | 1.46 | 7354.20 | 23.20 | 1.27 | 6413.08 | 73.82 | 1.11 |
| 20 | 720 | 6845.96 | 208.28 | 7497.00 | 14.33 | 1.10 | 10,058.92 | 32.17 | 1.47 | 8815.76 | 36.72 | 1.29 | 7598.60 | 123.75 | 1.11 |
| 20 | 840 | 7990.16 | 211.68 | 8720.12 | 15.65 | 1.09 | 11,563.44 | 44.00 | 1.45 | 10,249.36 | 55.19 | 1.28 | 8785.20 | 192.60 | 1.10 |
| 20 | 960 | 8990.68 | 214.96 | 10,092.48 | 16.42 | 1.12 | 13,098.08 | 61.37 | 1.46 | 11,683.08 | 86.68 | 1.30 | 9837.24 | 314.01 | 1.09 |
| 20 | 1080 | 10,021.92 | 220.11 | 11,400.92 | 17.09 | 1.14 | 14,726.88 | 72.97 | 1.47 | 13,069.84 | 107.84 | 1.30 | 10,933.88 | 402.99 | 1.09 |
| 20 | 1200 | 11,082.84 | 225.07 | 12,698.28 | 17.50 | 1.15 | 16,553.00 | 90.39 | 1.46 | 14,369.32 | 138.40 | 1.30 | 12,015.16 | 537.30 | 1.08 |
| C  | N   | AP3+VPG | ILS          | Seed          | CWI          | CWII         |
|----|-----|---------|--------------|---------------|--------------|--------------|
|    |     | Obj | CPU | Obj | CPU | Obj ratio | Obj | CPU | Obj ratio | Obj | CPU | Obj ratio |
| 10 | 480 | 5466.08 | 201.72 | 6492.20 | 3.74 | 1.19 | 8537.80 | 3.65 | 1.56 | 7150.08 | 6.54 | 1.31 | 6471.24 | 22.60 | 1.18 |
| 600 | 6785.28 | 202.71 | 8014.80 | 5.25 | 1.18 | 10,548.72 | 5.63 | 1.55 | 8635.08 | 12.62 | 1.27 | 7852.76 | 42.92 | 1.16 |
| 720 | 8122.40 | 204.05 | 9803.16 | 5.06 | 1.21 | 12,795.60 | 8.32 | 1.58 | 10,297.88 | 23.49 | 1.28 | 9224.68 | 81.00 | 1.14 |
| 840 | 9294.16 | 205.89 | 11,347.40 | 5.59 | 1.22 | 14,584.96 | 11.56 | 1.58 | 11,874.44 | 40.89 | 1.31 | 10,561.44 | 136.98 | 1.14 |
| 960 | 10,658.40 | 208.17 | 13,117.68 | 6.13 | 1.23 | 16,625.32 | 15.32 | 1.57 | 13,495.08 | 70.22 | 1.31 | 12,007.52 | 212.92 | 1.13 |
| 1080 | 11,844.72 | 211.34 | 14,852.72 | 6.27 | 1.25 | 18,575.56 | 20.43 | 1.57 | 15,054.52 | 109.72 | 1.27 | 13,364.36 | 339.19 | 1.13 |
| 1200 | 13,129.44 | 214.95 | 16,693.48 | 6.60 | 1.27 | 20,677.66 | 26.40 | 1.57 | 16,708.56 | 140.83 | 1.27 | 14,729.76 | 492.95 | 1.12 |
| 15 | 480 | 4386.08 | 202.11 | 5294.04 | 4.81 | 1.19 | 6849.88 | 4.45 | 1.56 | 5838.08 | 6.69 | 1.33 | 5217.20 | 24.46 | 1.19 |
| 600 | 5448.84 | 203.27 | 6668.04 | 5.34 | 1.22 | 8528.96 | 6.84 | 1.57 | 7141.72 | 12.07 | 1.31 | 6392.40 | 47.74 | 1.17 |
| 720 | 6465.56 | 204.96 | 8012.76 | 6.11 | 1.24 | 10,218.36 | 10.24 | 1.58 | 8601.36 | 22.53 | 1.33 | 7572.92 | 89.82 | 1.17 |
| 840 | 7632.00 | 207.13 | 9408.36 | 6.85 | 1.24 | 11,742.72 | 13.99 | 1.58 | 9995.88 | 39.02 | 1.32 | 8840.40 | 149.58 | 1.17 |
| 960 | 8508.92 | 210.45 | 10,871.52 | 6.57 | 1.28 | 13,312.44 | 19.68 | 1.56 | 11,224.16 | 79.27 | 1.32 | 9887.76 | 254.46 | 1.16 |
| 1080 | 9653.24 | 213.84 | 12,197.40 | 7.56 | 1.28 | 14,927.44 | 25.15 | 1.55 | 12,442.40 | 119.97 | 1.29 | 10,712.12 | 388.92 | 1.11 |
| 1200 | 10,510.24 | 216.40 | 13,608.76 | 7.98 | 1.29 | 16,352.12 | 29.64 | 1.57 | 13,626.96 | 130.82 | 1.30 | 11,812.36 | 489.23 | 1.12 |
| 16 | Test(800) | 6903.20 | 206.89 | 8138.80 | 10.23 | 1.18 | 10,771.00 | 16.14 | 1.57 | 9278.00 | 68.60 | 1.34 | 7484.00 | 144.19 | 1.08 |
| 20 | 480 | 3715.32 | 202.34 | 4500.00 | 6.10 | 1.24 | 5781.76 | 4.99 | 1.57 | 5060.60 | 6.15 | 1.36 | 4456.84 | 24.33 | 1.20 |
| 600 | 4470.80 | 203.92 | 5671.84 | 6.97 | 1.24 | 7177.08 | 7.91 | 1.57 | 6288.20 | 15.41 | 1.38 | 5436.32 | 52.37 | 1.19 |
| 720 | 5247.88 | 206.05 | 6852.88 | 7.18 | 1.26 | 8530.28 | 11.68 | 1.57 | 7447.08 | 23.70 | 1.37 | 6559.68 | 94.80 | 1.21 |
| 840 | 5824.72 | 208.35 | 8058.04 | 8.22 | 1.27 | 9864.43 | 16.22 | 1.56 | 8505.64 | 37.80 | 1.35 | 7438.88 | 156.15 | 1.18 |
| 960 | 7142.36 | 211.84 | 9292.48 | 8.81 | 1.30 | 11,196.84 | 22.16 | 1.57 | 9763.28 | 69.08 | 1.37 | 8459.68 | 271.83 | 1.18 |
| 1080 | 8037.04 | 216.31 | 10,573.92 | 8.65 | 1.32 | 12,510.04 | 28.44 | 1.57 | 10,865.72 | 107.20 | 1.35 | 9425.56 | 391.83 | 1.17 |
| 1200 | 8812.24 | 219.35 | 11,831.84 | 8.72 | 1.34 | 13,812.16 | 33.92 | 1.57 | 11,878.52 | 150.96 | 1.35 | 10,277.00 | 497.41 | 1.17 |
7. Conclusion

This study focuses on a variant of the integrated order batching problem that takes the packing operation in warehouse management into account and presents a solution approach. The integrated problem is modelled based on a real-world case, aiming to reduce the overall labour costs of both picking and packing processes by determining the assignment of orders in each batch. To tackle the complexity of the integrated problem, a statistical-based framework is proposed, defining a class of approximate models with the purpose of identifying the one with the highest correlation with the exact model. Subsequently, a hybrid algorithm is proposed, which combines an approximate model generated and selected within the framework and a heuristic that is based on the relationship between the number of visited aisles and the objective value revealed by the framework. The experimental results validate the effectiveness of the framework and the efficiency of the hybrid algorithm for a variety of different scales in practical scenarios, compared with other algorithms, including both integrated and picking-only methods. Thus, we can conclude that the discussed usage of this integrated model and solution method can effectively reduce the total labour costs of the picking and packing process.

In the future, other related operations in the warehouse, such as delivery, can be incorporated into the integrated problem. Additionally, the statistical-based framework can be extended to other problems to tackle the complexity introduced by the integration of multiple operations.

Data availability statement

The data that support the findings of this study are available from the corresponding author Chuanhou Gao, upon reasonable request.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was funded by the National Nature Science Foundation of China [grant numbers 12071428, 62111530247 and 12320101001], and the Zhejiang Provincial Natural Science Foundation of China [grant number LZ20A010002].

Notes on contributors

Sen Xue received the B.S. degree in Information and Computing Science from Zhejiang University, Hangzhou, China in 2020. He is currently working toward the Ph.D. degree in Operations Research in the School of Mathematical Sciences, Zhejiang University, Hangzhou, China. His research interests include Integer Programming, machine learning and logistics.

Chuanhou Gao received the B.Sc. degrees in Chemical Engineering from Zhejiang University of Technology, China, in 1998, and the Ph.D. degrees in Operational Research and Cybernetics from Zhejiang University, China, in 2004. From June 2004 until May 2006, he was a Postdoctor in the Department of Control Science and Engineering at Zhejiang University. Since June 2006, he has joined the Department of Mathematics at Zhejiang University, where he is currently a Full Professor. He was a visiting scholar at Carnegie Mellon University from Oct. 2011 to Oct. 2012. His research interests are in the areas of optimisation, chemical reaction network theory, machine learning, and thermodynamic process control. He is an associate editor of IEEE Transactions on Automatic Control and of International Journal of Adaptive Control and Signal Processing.

ORCID

Chuanhou Gao http://orcid.org/0000-0001-9030-2042

References

Abolfotoh, Aaya, Manjeet Singh, and Gursel Suer. 2019. “Order Batching Optimization for Warehouses with Cluster-Picking.” Procedia Manufacturing 39: 1464–1473. https://doi.org/10.1016/j.promfg.2020.01.302.

Ackerman, Kenneth B. 2012. Practical Handbook of Warehousing. New York: Springer Science & Business Media.

Akkerman, Fabian, and Martijn Mes. 2022. “Distance Approximation to Support Customer Selection in Vehicle Routing Problems.” Annals of Operations Research. https://doi.org/10.1007/s10479-022-04674-8.
Figliozzi, Miguel Andres. 2007. “Analysis of the Efficiency of Urban Commercial Vehicle Tours; Data Collection, Methodology, and Policy Implications.” Transportation Research Part B: Methodological 41 (9): 1014–1032. https://doi.org/10.1016/j.trb.2007.04.006.

Figliozzi, Miguel Andres. 2008. “Planning Approximations to the Average Length of Vehicle Routing Problems with Varying Customer Demands and Routing Constraints.” Transportation Research Record 2089 (1): 1–8. https://doi.org/10.3141/2089-01.

Franceschetti, Anna, Ola Jabali, and Gilbert Laporte. 2017. “Continuous Approximation Models in Freight Distribution Management.” Top 25 (3): 413–433. https://doi.org/10.1007/s11750-017-0456-1.

Gademann, Noud, and Steef Veld. 2005. “Order Batching to Minimize Total Travel Time in a Parallel-Aisle Warehouse.” IIE Transactions 37 (1): 63–75. https://doi.org/10.1080/074081705016917.

Gallien, Jérémie, and Théophane Weber. 2010. “To Wave Or Not to Wave? Order Release Policies for Warehouses with an Automated Sorter.” Manufacturing & Service Operations Management 12 (4): 642–662. https://doi.org/10.1287/msom.1100.0291.

Gasse, Maxime, Didier Chetelat, Nicola Ferroni, Laurent Charlin, and Andrea Lodi. 2019. “Exact Combinatorial Optimization with Graph Convolutional Neural Networks.” In Advances in Neural Information Processing Systems, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Vol. 32. Curran Associates, Inc.

Gibson, David R, and Gunter P Sharp. 1992. “Order Batching Procedures.” European Journal of Operational Research 58 (1): 57–67. https://doi.org/10.1016/0377-2217(92)90235-2.

Gu, Jinxiang, Marc Goetschalckx, and Leon F. McGinnis. 2007. “Research on Warehouse Operation: A Comprehensive Review.” European Journal of Operational Research 177 (1): 1–21. https://doi.org/10.1016/j.ejor.2006.02.025.

Härdle, Wolfgang, and Léopold Simar. 2007. Multivariate Statistical Analysis, Vol. 22007. Berlin, Heidelberg: Springer.

Henn, Sebastian, Sören Koch, Karl F Doerner, Christine Strauss, and Gerhard Wäscher. 2010. “Metaheuristics for the Order Batching Problem in Manual Order Picking Systems.” Business Research 3 (1): 82–105. https://doi.org/10.1007/bf03342717.

Henn, Sebastian, and Gerhard Wäscher. 2012. “Tabu Search Heuristics for the Order Batching Problem in Manual Order Picking Systems.” European Journal of Operational Research 222 (3): 484–494. https://doi.org/10.1016/j.ejor.2012.05.049.

Ho, Ying-Chin, Teng-Sheng Su, and Zhi-Bin Shi. 2008. “Order-Batching Methods for an Order-Picking Warehouse with Two Cross Aisles.” Computers & Industrial Engineering 55 (2): 321–347. https://doi.org/10.1016/j.cie.2007.12.018.

Ho, Y-C, and Y-Y Tseng. 2006. “A Study on Order-Batching Methods of Order-Picking in a Distribution Centre with Two Cross-aisles.” International Journal of Production Research 44 (17): 3391–3417. https://doi.org/10.1080/00207540600558015.

Hompel, Michael Ten, and Thorsten Schmidt. 2007. Warehouse Management. Berlin, Heidelberg: Springer.

Hong, Soondo, and Youngjoo Kim. 2017. “A Route-Selecting Order Batching Model with The S-Shape Routes in a Parallel-Aisle Order Picking System.” European Journal of Operational Research 257 (1): 185–196. https://doi.org/10.1016/j.ejor.2016.07.017.

Huang, Zeren, Kerong Wang, Furui Liu, Hui-Ling Zhen, Weinan Zhang, Mingxuan Yuan, Jianye Hao, Yong Yu, and Jun Wang. 2022. “Learning to Select Cuts for Efficient Mixed-Integer Programming.” Pattern Recognition 123: 108353. https://doi.org/10.1016/j.patcog.2021.108353.

Jiang, Wei, Jiyun Liu, Yun Dong, and Li Wang. 2021. “Assignment of Duplicate Storage Locations in Distribution Centres to Minimise Walking Distance in Order Picking.” International Journal of Production Research 59 (15): 4457–4471. https://doi.org/10.1080/00207543.2020.1766714.

Jiang, Xiaowei, Lijun Sun, Yuankai Zhang, and Xiangpei Hu. 2022. “Order Batching and Sequencing for Minimising the Total Order Completion Time in Pick-and-Sort Warehouses.” Expert Systems with Applications 187: 115943. https://doi.org/10.1016/j.epswa.2021.115943.

Li, Jundong, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. 2017. “Feature Selection: A Data Perspective.” ACM Computing Surveys (CSUR) 50 (6): 1–45. https://doi.org/10.1145/3136625.

Matusiak, Marek, René De Koster, and Jari Saarinen. 2017. “Utilizing Individual Picker Skills to Improve Order Batching in a Warehouse.” European Journal of Operational Research 263 (3): 888–899. https://doi.org/10.1016/j.ejor.2017.05.002.

Merchán, Daniel, and Matthias Winkenbach. 2019. “An Empirical Validation and Data-Driven Extension of Continuum Approximation Approaches for Urban Route Distances.” Networks 73 (4): 418–433. https://doi.org/10.1002/net.21874.

Ming-Huang Chiang, David, Chia-Ping Lin, and Mu-Chen Chen. 2014. “Data Mining Based Storage Assignment Heuristics for Travel Distance Reduction.” Expert Systems 31 (1): 81–90. https://doi.org/10.1111/exsy.12006.

Morabit, Mouad, Guy Desaulniers, and Andrea Lodi. 2021. “Machine-Learning–Based Column Selection for Column Generation.” Transportation Science 55 (4): 815–831. https://doi.org/10.1287/trsc.2021.1045.

Muter, Ibrahim, and Temel Öncan. 2015. “An Exact Solution Approach for the Order Batching Problem.” IIE Transactions 47 (7): 728–738. https://doi.org/10.1080/0740817x.2014.991478.

Nair, Vinod, Sergey Bartunov, Felix Gimeno, Ingrid von Glehn, Pawel Lichocki, Ivan Lobov, Brendan O’Donoghue, et al. 2020. “Solving Mixed Integer Programs Using Neural Networks.” https://arxiv.org/abs/2012.13349.

Nicola, Daniel, Rudolf Vetschera, and A Dragomir. 2019. “Total Distance Approximations for Routing Solutions.” Computers & Operations Research 102: 67–74. https://doi.org/10.1016/j.cor.2018.10.008.

Paulus, Max B, Giulia Zarrellon, Andreas Krause, Laurent Charlin, and Chris Maddison. 2022. “Learning to Cut by Looking Ahead: Cutting Plane Selection via Imitation Learning.” In Proceedings of the 39th International Conference on Machine Learning, edited by Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, Vol. 162. PMLR. https://proceedings.mlr.press/v162/paulus22a.html.

Rhee, WanSoO T., and Michel Talagrand. 1989. “A Sharp Deviation Inequality for the Stochastic Traveling Salesman Problem.” The Annals of Probability 17 (1). https://doi.org/10.1214/aop/1176991490.
Shiau, Jiun-Yan, and Ming-Chang Lee. 2010. “A Warehouse Management System with Sequential Picking for Multi-Container Deliveries.” *Computers & Industrial Engineering* 58 (3): 382–392. https://doi.org/10.1016/j.cie.2009.04.017.

Steele, J. Michael. 1981. “Complete Convergence of Short Paths and Karp’s Algorithm for the TSP.” *Mathematics of Operations Research* 6 (3): 374–378. https://doi.org/10.1287/moor.6.3.374.

Tang, Yunhao, Shipra Agrawal, and Yuri Faenza. 2020. “Reinforcement Learning for Integer Programming: Learning to Cut.” In *Proceedings of the 37th International Conference on Machine Learning*, edited by Hal Daumé III and Aarti Singh, Vol. 119. 9367–9376. PMLR. https://proceedings.mlr.press/v119/tang20a.html.

Tang, Lixin, Gongshu Wang, Jijin Liu, and Jingyi Liu. 2011. “A Combination of Lagrangian Relaxation and Column Generation for Order Batching in Steelmaking and Continuous-Casting Production.” *Naval Research Logistics (NRL)* 58 (4): 370–388. https://doi.org/10.1002/nav.20452.

Tappia, Elena, Debjit Roy, Marco Melacini, and René De Koster. 2019. “Integrated Storage-Order Picking Systems: Technology, Performance Models, and Design Insights.” *European Journal of Operational Research* 274 (3): 947–965. https://doi.org/10.1016/j.ejor.2018.10.048.

Valle, Cristiano Arbex, John E Beasley, and Alexandre Salles Da Cunha. 2017. “Optimally Solving the Joint Order Batching and Picker Routing Problem.” *European Journal of Operational Research* 262 (3): 817–834. https://doi.org/10.1016/j.ejor.2017.03.069.

Van Gils, Teun, An Caris, Katrien Ramaekers, Kris Braeckers, Benoit Depaire, and An Caris. 2018. “Increasing Order Picking Efficiency by Integrating Storage, Batching, Zone Picking, and Routing Policy Decisions.” *International Journal of Production Economics* 197: 243–261. https://doi.org/10.1016/j.ijpe.2017.11.021.

Wang, Meng, Ren-Qian Zhang, and Kun Fan. 2020. “Improving Order-Picking Operation Through Efficient Storage Location Assignment: A New Approach.” *Computers & Industrial Engineering* 139: 106186. https://doi.org/10.1016/j.cie.2019.106186.

Wolsey, Laurence A. 2020. *Integer Programming*. Hoboken: John Wiley & Sons.

Zare Mehrjerdi, Y, M Alipour, and A Mostafaeipour. 2018. “Integrated Order Batching and Distribution Scheduling in a Single-Block Order Picking Warehouse Considering S-Shape Routing Policy.” *International Journal of Engineering* 31 (10): 1723–1733.

Zhang, Jun, Xuping Wang, Felix TS Chan, and Junhu Ruan. 2017. “On-Line Order Batching and Sequencing Problem with Multiple Pickers: A Hybrid Rule-Based Algorithm.” *Applied Mathematical Modelling* 45: 271–284. https://doi.org/10.1016/j.apm.2016.12.012.

Zhong, Shuya, Vaggelis Giannikas, Jorge Merino, Duncan McFarlane, Jun Cheng, and Wei Shao. 2022. “Evaluating the Benefits of Picking and Packing Planning Integration in E-Commerce Warehouses.” *European Journal of Operational Research* 301 (1): 67–81. https://doi.org/10.1016/j.ejor.2021.09.031.

Zulj, Ivan, Sergej Kramer, and Michael Schneider. 2018. “A Hybrid of Adaptive Large Neighborhood Search and Tabu Search for the Order-Batching Problem.” *European Journal of Operational Research* 264 (2): 653–664. https://doi.org/10.1016/j.ejor.2017.06.056.