Improved formulations of the joint order batching and picker routing problem

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

ABSTRACT
Order picking is the process of retrieving ordered products from storage locations in warehouses. In picker-to-parts order picking systems, two or more customer orders may be grouped and assigned to a single picker. Then routing decisions regarding the visiting sequence of items during a picking tour must be made. Won and Olafsson (2005) found that solving the integrated problem of batching and routing enables warehouse managers to organize order picking operations more efficiently compared with solving the two problems separately and sequentially. We therefore investigate the mathematical programming formulation of this integrated problem.

We present several improved formulations of the problem based on the findings of Valle, Beasley, and Salles da Cunha (2017), that can significantly improve computational results. More specifically, we reconstruct the connectivity constraints and generate new cutting planes in our branch-and-cut framework. We also discuss some problem properties by studying the structure of a graphical representation of the warehouse, and we present two types of additional constraints. We also consider the no-reversal case of this problem. We present efficient formulations by building different auxiliary graphs. Finally, we present computational results for publicly available test problems for single-block and multiple-block warehouse configurations.

ARTICLE HISTORY
Received 28 January 2022
Accepted 26 October 2022

KEYWORDS
Integer programming; inventory management; order batching; order picking; picker routing

1. Introduction
In modern business environments, warehousing and relative order picking processes are essential components of any supply chain (Manzini 2012). Order picking is the process of retrieving ordered products from storage locations in warehouses, and it typically accounts for 55% of the total warehouse operating expense and has long been recognized as the most labor-intensive and costly activity for warehouses. Therefore, the order picking process should be robustly designed and optimally controlled to handle requirements efficiently (de Koster, Le-Duc, and Jan Roodbergen 2007; Tompkins et al. 2010).

The order picking process should be investigated within a system context. In picker-to-parts systems, pickers walk or ride through the picking area to collect the requested items. In parts-to-picker systems, automated cranes move along the aisle, retrieve unit loads, and bring them to a pick position. This study focuses on order picking operations in picker-to-parts systems, which still account for the large majority of all order picking systems (de Koster, Le-Duc, and Jan Roodbergen 2007; Marchet, Melacini, and Perotti 2015; van Gils et al. 2018).

According to (van Gils et al. 2018), decisions to manage order picking in picker-to-parts systems can be classified into strategic, tactical or operational decisions. In this paper we focus on the operational planning problems that typically concern daily operations. Operational planning problems include (1) how batches of orders are generated (batching), (2) how each picker is routed (routing), (3) how the daily required number of order pickers is determined (workforce level), (4) how a given workforce is allocated to the picking and sorting operations (workforce allocation), and (5) how the picking orders are sequenced (job assignment). Some researchers focus on individual planning problems, have developed many efficient algorithms, and have successfully applied these algorithms to these planning problems. However, simultaneously optimizing multiple order picking planning problems can result in even more efficient picking operations. van Gils et al. (2018) presented a comprehensive review of research on combination problems in order picking systems. According to their study, the joint order batching and picker routing problem (JOBPRP), which is also the focus of attention in this study, has received more attention than other combination problems (for example, the integrated problem of order batching and job assignment). We now introduce the JOBPRP from practical application viewpoints. In picker-to-parts systems, an order picker has to pick one or more customer orders on his/her picking tour. A customer order typically
requires a list of distinct products, and customer orders can be grouped into a picking tour if the capacity of the picking device is not exhausted. To minimize the total travel distance, the warehouse management system has to decide how the orders should be assigned to picking tours, which gives rise to the so-called order batching problem. For each picking tour, the shortest path to visit a set of picking locations should be determined, which gives rise to the so-called picker routing problem. Those two problems are thought to be strongly linked, because solving the routing problem is dependent on the solution of the batching problem. Obviously, solving the integrated problem, the JOBPRP, can improve the efficiency of order pickers. One of the first works that considers this integrated problem was (Won and Olafsson 2005). The crucial observation from their simulation experiment is that a simultaneous solution yields significantly better performance than a sequential solution. More recently, Scholz and Wäscher (2017) integrated different routing algorithms into a heuristic approach for the batching problem. Their numerical experiments demonstrated the benefits from solving the joint problem.

Although there are many heuristic-solving frameworks, very few exact algorithms for the JOBPRP have been proposed in the literature. Valle, Beasley, and Salles da Cunha (2017) proposed a novel exact algorithm that incorporates a non-compact integer programming formulation and a branch-and-cut procedure. The original model proposed by Valle, Beasley, and Cunha (2016) can be significantly improved by adding valid inequalities based on the standard layout of warehouses. Inspired by Valle, Beasley, and Salles da Cunha (2017), we analyze existing models for the JOBPRP and propose modified solution approaches to improve computational performance. The following are the main contributions of our study:

1. We reconstruct the connectivity constraints (which enforce connectivity requirements for closed walks) by making full use of the properties of a rectangular warehouse with multiple blocks, and the improved formulations can be solved with high efficiency in the experiments.

2. We also discuss some problem properties by studying the structure of a graphical representation of the warehouse, and we present two types of additional constraints.

3. We also consider the no-reversal JOBPRP, where a picker can only reverse direction at the end of a subaisle. We propose a traveling salesman problem (TSP) formulation for single-block and 2-block warehouses by introducing two auxiliary graphs.

4. We conduct a series of numerical experiments to evaluate our formulations. The test instances used here are generated using the method provided by (Valle, Beasley, and Cunha 2016).

The remainder of this paper is organized as follows. Section 2 comprises a literature review regarding the JOBPRP and some closely related problems. Section 3 briefly introduces the JOBPRP and presents a graph-based formulation for this problem. Section 4 presents two improved formulations, and Section 5 introduces two types of additional constraints to improve solution quality. Section 6 discusses the no-reversal JOBPRP in detail. Section 7 reports on some computational results and observations, and Section 8 concludes the study.

2. Literature review

Picker routing and order batching problems have received considerable interest since the 1980s (Elsayed 1981; Andrew, Vannelli, and Kumar 1986; Ratliff and Rosenthal 1983; Cornuéjols, Fonlupt and Naddef 1985). In this section, we provide a summary of some previous studies on order batching and picker routing problems.

2.1. Picker routing problem

This problem can be solved to optimality using any exact approach to the TSP. However, more efficient solution approaches can be obtained using a particular warehouse layout. The first attempt to provide a problem-specific exact approach to picker routing problems was proposed by Ratliff and Rosenthal (1983). They constructed a sparse graph representation for a rectangular warehouse containing a single block and presented a polynomial-time dynamic programming algorithm. Researchers then developed two generalized algorithms based on the work of Ratliff and Rosenthal (1983). Cornuéjols, Fonlupt and Naddef (1985) interpreted the routing problem as a Steiner travelling salesman problem (STSP) and extended the Ratliff-Rosenthal algorithm to all series-parallel graphs; Roodbergen and Koster (2001) modified the Ratliff-Rosenthal algorithm and introduced routing heuristics for 2-block and more complex layouts. Recently, Scholz et al. (2016) proposed an exact approach regarding the unique structure of a single-block warehouse. They introduced an integer programming model whose size is independent of the number of picking locations and demonstrated that this formulation can significantly improve computational performance.

Because the TSP is NP-hard, optimal routing is often regarded as difficult to determine. Many heuristics have been proposed for the problem from a
practical standpoint based on different routing strategies, including S-shape (Goetschalckx and Ratliff 1988), midpoint (Hall 1993), largest gap (Hall 1993), combined (Petersen 1997) and aisle-by-aisle (Vaughan 1999). TSP heuristics can also be used to solve the picker routing problem. Theys et al. (2010) discovered that the Lin–Kernighan–Helsgaun heuristic (Helsgaun 2000) outperforms the S-shape heuristic when there are two or more blocks in the warehouse. Cambazard and Catusse (2018) developed a dynamic programming approach for a rectilinear TSP, and the algorithm is also applicable to the picker routing problem. However, the complexity grows exponentially with the number of blocks. Other heuristics are created from metaheuristics. For example, Ho and Tseng (2006) proposed a simulating annealing heuristic which is integrated with the largest gap routing strategy. The reader interested in the picker routing problem may refer to Masae, Glock, and Grosse (2020) for a comprehensive review.

2.2. Joint order batching and picker routing problem

The order batching problem can be formally defined as follows: How, given the capacity of the picking device and the adopted routing strategy, can a given set of customer orders with known storage locations be grouped into picking tours such that the total length of all picker tours is minimized? (Wäscher 2004; Manzini 2012)

Because the order batching problem is known to be NP-hard (Gademann, Berg, and Hoff 2001), exact approaches are typically impractical for instances with relatively large sizes. As a result, many scholars consider heuristics and metaheuristics. According to Manzini (2012), batching heuristics can be distinguished into savings, seed, or priority rule-based algorithms as well as other algorithms. The interested reader may refer to Manzini (2012), van Gils et al. (2018), and Cergibozan and Tasan (2019) for further details.

Considering the strong relationship between batching and routing, solving these planning problems in an integrated manner would be beneficial. In recent decades, many efficient heuristic and metaheuristic methods have been proposed to solve the JOBPRP. Won and Olafsson (2005) was one of the first to formulate the batching and routing problem jointly as a combinatorial optimization problem. Their proposed two-step heuristic first constructs batches sequentially and then solves the subsequent routing problem. Hong, Johnson, and Peters (2012) presented a route-selection based formulation that enumerates all possible routes and compared their heuristic solution with a lower bound developed by a relaxation model. Kulak, Sahin, and Taner (2012) proposed a tabu search algorithm integrated with a clustering algorithm that generates an initial solution. They also proposed two constructive heuristics to solve the picker routing problem. Grosse, Glock, and Ballester-Ripoll (2014) developed a simulated annealing algorithm to determine order batches and picker routes and applied four different heuristics to form initial order batches. Cheng et al. (2015) proposed a hybrid approach consisting of a particle swarm optimization for batching, whereas (Li, Huang, and Dai 2017) proposed a constructive heuristic based on similarity coefficient for batching; both used an ant colony optimization algorithm in the routing procedure. Scholz and Wäscher (2017) introduced an iterated local search algorithm which allows for integrating different routing algorithms, and (Arbex Valle and Beasley 2020) presented an approximate formulation for this problem; they also proposed a partial integer optimization heuristic based on their formulation. Brant et al. (2020) proposed a heuristic based on column generation to deal with an exponential linear programming formulation of the JOBPRP. Aerts, Cornelissens, and Sörensen (2021) modeled the JOBPRP as a clustered vehicle routing problem and applied a two-level variable neighborhood search algorithm developed by Defryn and Sörensen (2017). Attari et al. (2021) presented a model of JOBPRP under uncertainty and used metaheuristics such as genetic, particle swarm optimization, and artificial honeybee colony algorithms to solve the formulated model.

To enhance efficiency and customer service, some researchers also take the due dates of the customer orders into account, which initiates the picking sequencing problem. Tsai, Liou, and Huang (2008) considered earliness and tardiness penalties and suggested a genetic algorithm under the assumption that splitting customer orders is allowed. Chen et al. (2015) developed a genetic algorithm for the order batching and sequencing processes. For the routing decision for each batch, they adopted an ant colony algorithm. André, Schubert, and Wäscher (2017) introduced an iterated local search algorithm to solve the problem effectively and efficiently. They also showed the substantial performance benefits gained from integrating planning problems using a real-life case study.

Apart from these heuristic and metaheuristic methods, only a few exact approaches have been proposed in the literature. Valle, Beasley, and Cunha (2016) presented three basic formulations of the JOBPRP; one of them involves exponentially many constraints and
the remaining two are based on network flows. They used the branch-and-cut algorithm presented by Padberg and Rinaldi (1991) for the first formulation. A JOBPRP-test instance generator based on publicly available real-world data was also introduced. The non-compact formulation proposed by Valle, Beasley, and Cunha (2016) was improved by Valle, Beasley, and Salles da Cunha (2017). They introduced a significant number of valid inequalities to strengthen the linear relaxation of their formulation.

3. Problem description and basic formulation

In this section, we provide more background information regarding the warehouse layout that we consider. We then introduce a basic formulation of the JOBPRP.

3.1. Background information

We consider a rectangular warehouse with a manual picker-to-parts order-picking system. The warehouse is composed of one origin, several vertical picking aisles, and several horizontal cross-aisles. We call the part between two adjacent cross-aisles a block, and the section of a picking aisle within a block a subaisle. If a warehouse has \( q \) blocks, any picking aisle in the warehouse can be partitioned into \( q \) subaisles. We illustrate these concepts in Figure 1.

Each vertical aisle contains a set of picking locations on both sides and a picking location contains several storage slots. We assume that each slot holds one type of product and each product type is assigned to only one slot. In addition, products are divided into several classes, and products belonging to the same class are placed in consecutive slots. As a result, most items in a subaisle belong to the same class. The adopted storage assignment policy is actually a variant of the class-based storage policy (Manzini 2012).

Before a shift starts, the number of order pickers available for carrying out the picking operations has been determined. Pickers start at the origin, visit a set of picking locations to retrieve the ordered products and return to the origin. During a picking tour, a picker is equipped with a trolley that accommodates a limited number of baskets. The necessary number of baskets to carry each customer order is assumed to be known. A customer order typically requires a list of distinct products, and we assume that a customer order cannot be split over various batches. The reason is that mixing and dividing orders can result in an unacceptable consolidation effort (Valle, Beasley, and Cunha 2016; Scholz and Wäscher 2017).

3.2. Model formulation

The basic formulation of the JOBPRP that we discuss in this section is based on the STSP. Thus we first introduce a formulation for the STSP in the context of the single-picker routing problem. Then, we provide the basic formulation for the JOBPRP.

Let \( V_L \) denote a set of picking locations, each of which is on a subaisle and contains one or more requested products. Let \( V_I \) denote a set of endpoints of each subaisle, and we call these points ‘artificial locations’. For simplicity, we assume that the origin \( s \) is located at the top left corner of the warehouse, and it just overlaps the first artificial location. We define a sparse graph using the vertex set \( V = V_L \cup V_I \) and the edge set \( E \). Edges in \( E \) connect the following pairs of vertices: (1) two neighboring locations within a picking aisle and (2) two neighboring artificial locations within a cross-aisle. The length \( d_e \) of any edge \( e \in E \) is equal to the direct distance between two locations. The graphical representation of a warehouse is shown in Figure 2.

Now we present a non-compact formulation, inspired by the work of Letchford, Nasiri, and Oliver Theis (2013), for the single-picker routing problem. Before a picker enters the warehouse, the graphical representation \( G = (V, E) \) is already known. The task is to find a closed walk by which every \( v \in V_L \) is visited. Note that the walk does not need to be Hamiltonian or Euler circuits, that is, a vertex or an edge can be visited more than once by the walk. Let \([u, v]\) denote the unordered pair of location \( u \) and location \( v \), i.e. \([u, v]\) is the edge connecting \( u \) and \( v \). For any node set \( S \subset V \), \( \delta(S) \) denotes the set of edges with exactly one end-node inside \( S \). For a single vertex \( v \in V \), let \( \delta(v) = \delta([v]) \). We introduce a nonnegative decision variable \( x_e \in \mathbb{Z} \) to represent the number of times

![Figure 1. Example of a 2-block warehouse layout.](image)
edge \( e \) is traversed. We also use a binary decision variable \( y_v \) to indicate whether vertex \( v \in V \setminus \{s\} \) is visited by the walk. The single-picker routing problem can be easily described by the following program.

\[
\begin{align*}
\min \quad & \sum_{e \in E} d_e x_e \\
\text{s.t.} \quad & \sum_{e \in \delta(v)} x_e \geq 1, \quad \forall v \in \{s\} \cup V_L \\
& y_v \geq \min\{x_v, 1\}, \quad \forall v \in V \setminus \{s\}, e \in \delta(v) \\
& \sum_{e \in \delta(S)} x_e \geq y_v, \quad \forall v \in S, S \subset V \setminus \{s\}, |S| \geq 2 \\
& \sum_{e \in \delta(v)} x_e \quad \text{is an even integer}, \quad \forall v \in V \\
& x_e \in \{0, 1, 2, \ldots\}, \quad \forall e \in E \\
& y_v \in \{0, 1\}, \quad \forall v \in V
\end{align*}
\]

Constraints (2) ensure that each picking location is visited by the picker. Constraints (3) define the \( y \) variables for each vertex. Constraints (4) guarantee that the multigraph induced by the walk is connected. In the TSP, constraints (4) are also known as subtour elimination constraints. Constraints (3), (4), and (5) ensure that an Euler circuit exists in the multigraph induced by the walk.

The abovementioned formulation is treated as a starting point for the JOBPRP. The basic formulation for the JOBPRP is constructed by including the assignment of orders to batches as additional constraints. We start by creating a directed graph \( \tilde{G} = (V, \tilde{E}) \) from graph \( G = (V, E) \): any edge \( e = (u, v) \in E \) is replaced with two directed arcs \( e_1 = (u, v) \) and \( e_2 = (v, u) \). A formulation based on graph \( \tilde{G} \) for the single-picker routing problem can be formulated similarly. Constraints (5) will then be replaced by flow constraints, which are known to be totally unimodular constraints. Furthermore, a very useful theorem proposed in (Valle, Beasley, and Cunha 2016) is as follows.

**Theorem 3.1:** Each directed arc in \( \tilde{E} \) can only be traversed once by any optimal walk.

When multiple pickers and pick capacity are considered, we should assign orders to pickers. Assume that a picker has a capacity of \( B \) units, the set of all orders is denoted by \( O \), and any order \( o \in O \) requires a capacity of \( b_o \) units. Two or more customer orders can be batched together if the total capacity of customer orders assigned to a picker does not exceed its available capacity. After the order batching process, \( O \) will be partitioned into several subsets, and each subset of orders will be assigned to one picker. Each order \( o \) contains a subset of picking locations \( L_o \subset V \). Let \( O_t \subset O \) denote the subset of orders assigned to picker \( t \), then picker \( t \) must visit all nodes in \( \bigcup_{o \in O_t} L_o \) to collect a set of items for orders in \( O_t \).

Now we present a basic formulation of the JOBPRP based on formulation (1)–(7) and Theorem 3.1. For any node set \( S \subset V \), let \( \delta^+(S) = \{(u, v) \in \tilde{E} : u \in S, v \not\in S\} \) and \( \delta^-(S) = \{(u, v) \in \tilde{E} : u \not\in S, v \in S\} \). For a single vertex \( v \in V \), let \( \delta^+(v) = \delta^+(\{v\}) \) and \( \delta^-(v) = \delta^-(\{v\}) \). Let \( T \) be the number of available pickers, and let \( T = \{1, 2, \ldots, T\} \). We introduce the binary variables \( x_{ltv} \) to indicate whether arc \( (u, v) \) is traversed by picker \( t \), \( y_{tv} \) to indicate whether vertex \( v \) is visited by picker \( t \) and \( z_{otv} \) to indicate whether picker \( t \) picks order \( o \). The basic formulation is formally given as follows.

\[
\begin{align*}
\min \quad & \sum_{t=1}^{T} \sum_{(u,v) \in \tilde{E}} d_{uv} x_{ltv} \\
\text{s.t.} \quad & \sum_{(s,v) \in \delta^+(s)} x_{ltv} \geq 1, \quad \forall t \in T \\
& \sum_{(u,v) \in \delta^+(u)} x_{ltv} \geq x_{ltu}, \quad \forall t \in T, \ o \in O, \ u \in L_o \\
& y_{tv} \geq x_{ltv}, \quad \forall t \in T, \ u \in V \setminus \{s\}, \ (u, v) \in \delta^+(u) \\
& \sum_{(u,v) \in \delta^+(S)} x_{ltv} \geq y_{ltv}, \quad \forall t \in T, \ S \subset V \setminus \{s\}, |S| \geq 2, \ u_0 \in S \\
& \sum_{(u,v) \in \delta^-(v)} x_{ltv} = \sum_{(u,v) \in \delta^-(v)} x_{ltu}, \quad \forall t \in T, \ v \in V
\end{align*}
\]
\[
\sum_{t \in T} z_{ot} = 1, \quad \forall o \in O \tag{14}
\]
\[
\sum_{o \in O} b_{o} z_{ot} \leq B, \quad \forall t \in T \tag{15}
\]
\[
x_{tv} \in \{0, 1\}, \quad \forall t \in T, \quad (u, v) \in \tilde{E} \tag{16}
\]
\[
y_{tv} \in \{0, 1\}, \quad \forall t \in T, \quad v \in V \tag{17}
\]
\[
z_{ot} \in \{0, 1\}, \quad \forall t \in T, \quad o \in O \tag{18}
\]

Constraints (9)–(13) are similar to constraints (2)–(5); each of them describes a picker routing process. Constraints (14) ensure that each order is assigned to precisely one picker. Constraints (15) are capacity constraints. Constraints (14)-(15) describe the order batching process. The feasible region of the basic formulation is denoted by \(P_{\text{basic}}\), i.e. \(P_{\text{basic}} = \{(x, y, z) : \text{constraints (9)–(18)}\}\). This formulation is non-compact because it involves exponentially many constraints (12) to enforce connectivity. A branch-and-cut algorithm that separates these constraints should be implemented when this formulation is adopted.

The basic formulation is a slightly different version of the original formulation (Valle, Beasley, and Cunha 2016). The original formulation assumes that it is unnecessary for a picker to depart from the origin when no order is assigned to it, but constraints (9) force all pickers to depart from the origin. The main reasons for this assumption are the following:

1. We do not tackle the workforce level planning problem in this paper, and this assumption helps in simplifying the formulation.
2. This is not a critical assumption because constraints (9) can easily be modified to deal with the previous assumption.

In the rest of the paper, we simply let \(|T|\) be the necessary number of pickers to carry all products, which can be obtained by solving a bin-packing problem.

4. Improved formulations

In this section, we propose two improved formulations \(P_G\) and \(P_F\) for the JOBPRP by reformulating the connectivity constraints (12). The key idea of reformulation is to enforce connectivity using the subaisle cuts introduced by Valle, Beasley, and Salles da Cunha (2017). To improve readability, we present two tables that summarize the most important notations and formulations in the appendix.

We first introduce the subaisle cuts and discuss the relationships between these cuts and the basic formulation. Let the number of subaisles be \(W_{\text{sub}}\). Subaisles are indexed by the elements in set \(W_{\text{sub}} = \{1, 2, \ldots, W_{\text{sub}}\}\). Let the set of picking locations within subaisle \(i\) be \(V_{\text{sub}}(i)\).

The northern artificial location is denoted by \(f(i)\) and the southern artificial location is denoted by \(l(i)\). For any picking location \(v \in V_{\text{sub}}(i)\), the adjacent northern location is denoted by \(n(v)\) and the adjacent southern location is denoted by \(s(v)\). \(s(f(i))\) and \(n(l(i))\) are defined similarly. By using auxiliary binary variables \(\alpha\) and \(\beta\), we consider the following feasible region \(P_{\text{sub}}\) containing only subaisle cuts:

\[
\alpha_{tv} \geq \alpha_{ts(v)}, \quad \forall t \in T, \quad i \in W_{\text{sub}}, \quad v \in V_{\text{sub}}(i) \\setminus \{n(l(i))\} \tag{19}
\]
\[
x_{ts(v)} v \geq \alpha_{tv}, \quad \forall t \in T, \quad i \in W_{\text{sub}}, \quad v \in V_{\text{sub}}(i) \tag{20}
\]
\[
\beta_{tv} \geq \beta_{tn(v)}, \quad \forall t \in T, \quad i \in W_{\text{sub}}, \quad v \in V_{\text{sub}}(i) \\setminus \{s(f(i))\} \tag{21}
\]
\[
x_{ts(v)} v \geq \beta_{tv}, \quad \forall t \in T, \quad i \in W_{\text{sub}}, \quad v \in V_{\text{sub}}(i) \tag{22}
\]
\[
\alpha_{tv} + \beta_{tv} \geq z_{ot}, \quad \forall t \in T, \quad o \in O, \quad v \in L_o \tag{23}
\]
\[
\alpha_{tv} \in \{0, 1\}, \quad \forall t \in T, \quad i \in W_{\text{sub}}, \quad v \in V_{\text{sub}}(i) \tag{24}
\]
\[
\beta_{tv} \in \{0, 1\}, \quad \forall t \in T, \quad i \in W_{\text{sub}}, \quad v \in V_{\text{sub}}(i) \tag{25}
\]

Those additional constraints work as follows: we suppose that there exists a feasible solution \((x^*, \alpha^*, \beta^*) \in P_{\text{sub}}\). For any picking location \(v\) in subaisle \(i\), if \(\alpha^*_{tv} = 1\) then there exists a straight path connecting \(f(i)\) and \(v\) in walk \(t\), and the same holds when \(\beta^*_{tv} = 1\). If walk \(t\) pass through picking location \(v\), then there must exist a path connecting \(v\) and the northern or southern artificial location. Therefore, constraints (23) are valid constraints. Remark that \(\alpha_{tv} + \beta_{tv}\) may not be 2 when there exist two paths in walk \(t\), one of them connects picking location \(v\) and the northern artificial location, and others connect \(v\) and the southern artificial location. In other words, there is no surjection from \(P_{\text{basic}}\) to \(P_A = \{(x, y, z, \alpha, \beta) : (x, y, z) \in P_{\text{basic}}, (x, z, \alpha, \beta) \in P_{\text{sub}}\}\).

Subaisle cuts can be regarded as a class of connectivity constraints. Therefore, those cuts can partially replace constraints (12). To illustrate the relationship between \(P_{\text{sub}}\) and constraints (12), consider the following example.

**Example 4.1:** Consider the warehouse illustrated in Figure 3. Suppose that there is an order \(o\) with \(L_o = \{v_{22}, v_{31}\}\). We provide feasible solutions for various relaxations of \(P_A\) in Figure 3:

1. When we remove all connectivity constraints, a feasible solution could only consist of several cycles.
(2) When we remove constraints (12), $v_{22}$ and $v_{31}$ are forced to be connected with neighboring artificial locations.

(3) When all connectivity constraints are used, we are able to generate a Eulerian tour from a feasible solution.

It is possible to observe that we only need to focus on the connectivity of the graph induced by artificial locations when subaisle cuts have been added to the formulation (Figure 3(c)). In the remainder of this section, we reconstruct constraints (12) using this observation. For any artificial location $v$, we let $Q_{W}(v)$ be the adjacent artificial location $u$ lying to the west of $v$; we let $Q_{E}(v)$ be the adjacent artificial location $u$ lying to the east of $v$. For any subaisle $i$, we let $Q_{N}(l(i)) = f(i)$ and $Q_{S}(f(i)) = l(i)$. We illustrate all of these concepts in the reduced graph in Figure 4. Let $\tilde{E}'$ be the arc set of the reduced graph, that is, $\tilde{E}'$ is a set of edges connecting neighboring artificial locations while ignoring picking locations within subaisles. For each arc $(u, v) \in \tilde{E}'$, we introduce an auxiliary binary variable $\gamma_{uv}$. For any node set $S \subset$
Theorem 4.2: \( \min \{ f(x) : (x, y, z, \alpha, \beta, \gamma) \in P_G \} = \min \{ f(x) : (x, y, z) \in P_{basic} \} \)

Proof: For some \((x^*, y^*, z^*) \in P_{basic}\), \((\alpha^*, \beta^*, \gamma^*)\) are defined as follows:

1. For any picking location \(v\) within subaisle \(i\), we set \(\alpha_i^* = 1\) if \(x_{tuv_i}^* = 1\) for \(u \in \{f(i), s(i), s(s(i))\}, \ldots, n(v)\); otherwise, we set \(\alpha_i^* = 0\).
2. For any picking location \(v\) within subaisle \(i\), we set \(\beta_i^* = 1\) if \(x_{tu(\gamma)} = 1\) for \(u \in \{l(i), n(l(i)), n(n(l(i))), \ldots, n(v)\}\); otherwise, we set \(\beta_i^* = 0\).
3. For any artificial location \(v\), we set \(\gamma_{tvQW(v)}^* = x_{tvQW(v)}^*\) and \(\gamma_{tvQW(v)}^* = x_{tvQW(v)}^*\). We set \(\gamma_{tvQW(v)}^* = 1\) if \(v = l(i)\) and \(x_{tu(\gamma)} = 1\) for \(u \in \{l(i), n(l(i)), \ldots, s(f(i))\}\); otherwise, we set \(\gamma_{tvQW(v)}^* = 0\). We define \(\gamma_{tvQW(v)}^*\) similarly.

It is easy to check that \((x^*, y^*, z^*, \alpha^*, \beta^*, \gamma^*) \in P_G\), therefore the following holds:

\[
\min \{ f(x) : (x, y, z, \alpha, \beta, \gamma) \in P_G \} \\
\leq \min \{ f(x) : (x, y, z) \in P_{basic} \}
\]
following result.
\[ \min \{ f(x) : (x, y, z, \alpha, \beta, \gamma) \in P_G \} \]
\[ \geq \min \{ f(x) : (x, y, z) \in P_{basic} \} \]

Proof: For some constraints (26)–(34), (36)–(42) and the following:
\[ \sigma_{fv}^{u0} = y_{tv0}, \quad \forall t \in T, \quad v_0 \in V_L \]
\[ \sum_{(u,v) \in \eta^+ (u)} \sigma_{uv}^{v0} - \sum_{(v,u) \in \eta^- (u)} \sigma_{uv}^{v0} = 0, \quad \forall t \in T, \quad v_0 \in V_L, \quad u \in V_L \setminus \{s, v_0\} \]
\[ \sum_{(v,s) \in \eta^- (s)} \sigma_{vs}^{v0} = -y_{tv0}, \quad \forall t \in T, \quad v_0 \in V_L \]
\[ 0 \leq \sigma_{tv}^{v0} \leq \gamma_{tau}, \quad \forall t \in T, \quad v_0 \in V_L, \quad (u,v) \in \tilde{E} \]

Let \( P_F \) = \{ (x, y, z, \alpha, \beta, \gamma, \sigma) : (x, y, z, \gamma, \sigma) \in P_f, (x, z, \alpha, \beta) \in P_{sub} \}. \) We immediately obtain the following.

Theorem 4.3: \( \min \{ f(x) : (x, y, z, \alpha, \beta, \gamma, \sigma) \in P_F \} = \min \{ f(x) : (x, y, z) \in P_{basic} \} \)

Proof: For some \( (x^*, y^*, z^*) \in P_{basic}, \) \( (\alpha^*, \beta^*, \gamma^*) \) are defined as in the proof of Theorem 4.2 and \( \sigma^* \) is defined as follows.

For picker \( t \in T, \) the graph indicated by \( x^* \) is denoted by \( H. \) For any artificial location \( v_0 \) with \( y_{tv0}^* = 1, \) we consider an arbitrary path in \( H \) that goes from \( v_0 \) to the origin (because \( (x^*, y^*, z^*) \in P_{basic} \), such a path exists). We set \( \sigma_{tvu}^{v0} = 1 \) if \( (u,v) \) is traversed by this path; otherwise, we set \( \sigma_{tvu}^{v0} = 0. \)

It is easy to check that \( (x^*, y^*, z^*, \alpha^*, \beta^*, \gamma^*, \sigma^*) \in P_F, \) and therefore the following holds:
\[ \min \{ f(x) : (x, y, z, \alpha, \beta, \gamma, \sigma) \in P_F \} \]
\[ \leq \min \{ f(x) : (x, y, z) \in P_{basic} \} \]

The rest of the proof is similar to that of Theorem 4.2.

We finish this section by showing the relationship between the LP relaxations of \( P_G \) and \( P_F. \) Formally, let \( P_{LP} \) denote the LP relaxation of \( P, \) let \( \text{proj}_x(P) \) denote the projection of \( P \) onto the \( x-space \) and we prove the following theorem:

Theorem 4.4: \( \text{proj}_{(x,y,z,\alpha,\beta,\gamma)}((P_F)_{LP}) = (P_G)_{LP} \)

Proof: According to [Theorem 2. Letchford, Nasiri, and Oliver Theis 2013], there exists a feasible flow \( \sigma_{u0}^{v0} \) satisfying constraints (43)–(46) if and only if
\[ \sum_{(u,v) \in \eta^+ (S)} y_{tvu} \geq y_{tu0}, \quad \forall S \subseteq \{S : u_0 \in S, S \subset V_L \setminus \{s\}\} \]

Because there should be a feasible flow \( \sigma_{u0}^{v0} \) for any \( u_0 \in V_L \setminus \{s\} \) according to our construction, the proof is ended.

5. Additional constraints

In this section, we introduce two types of additional constraints, called strengthened connectivity constraints and single traversing constraints. Strengthened connectivity constraints link the routing decisions and the batching decisions where classical connectivity constraints cannot take batching decisions into account. Single traversing constraints mainly focus on the problem property of a rectangular warehouse and can cut off some non-optimal solutions.

5.1. Strengthened connectivity constraints and basic cuts

Considering the relationship between routing decisions and batching decisions, it would be interesting to investigate the constraints that jointly deal with both decisions.
We first introduce the concept of the strengthened connectivity constraints. Let us have a look at connectivity constraints (12): a constraint of type (12) can be generated by fixing a proper vertex set $S$. For any picking location $u_0 \in S$, we know that $y_{u_0}$ is actually dominated by any $z_\omega$ satisfying $u_0 \in L_\omega$ according to constraints (10)-(11). Therefore, we have the following strengthened connectivity constraints:

$$\sum_{(u,v) \in \delta^+(S)} x_{uv} \geq z_\omega, \quad \forall \, t \in T, \, S \subset V \setminus \{s\}, |S| \geq 2, \quad o \in \{o : S \cap L_\omega \neq \emptyset \}$$ (47)

Notice that there are an exponential number of connectivity constraints (12). Thus, constraints (47) cannot be added directly to a formulation. To make use of these constraints, we intend to find a family $S$ of a polynomial number of vertex sets $S$. All constraints induced by $S$ will be added directly to a formulation. A carefully selected $S$ leads to powerful constraints. For instance, if $S \in S$ is composed of locations on the right side of some aisle (as shown in Figure 5), constraints (47) are reduced to aisle cuts proposed by Valle, Beasley, and Salles da Cunha (2017).

Here, we present another type of strengthened connectivity constraints, which we call basic cuts. $S$ can be obtained by the following operations. For any order $o$, let $E_0$ denote a set of $e \in \tilde{E}$ (which represents a subaisle) that contains at least one picking location $u \in L_\omega$. Let $V_0$ be a set of vertices associated with $E_0$. Then, any $e \in \tilde{E}$ with both ends in $V_0$ will be added to $E_0$. By executing a depth-first search algorithm, we can discover all connected components of the graph $(V_0, E_0)$. Figure 5 shows an example of the two connected components induced by an order. Let $S$ be a vertex set of a connected component that does not contain the origin. All picking locations within the subaisle that has both ends in $S$ will then be added to $S$. Then $S$ will be added to $S$. We call the strengthened connectivity constraints induced by $S$ basic cuts.

### 5.2. Single traversing constraints

In this subsection, we derive additional constraints by investigating the graph property of the warehouse. We first restrict our attention to a single-block warehouse. By ‘traversing’, we mean going from the north artificial vertex to the south artificial vertex or vice-versa.

**Theorem 5.1:** For a single-block warehouse, each subaisle will be traversed at most once by any optimal walk.

**Corollary 5.2 (Single traversing constraints):** For a single-block warehouse, no optimal solution will be cut off by the following constraints:

$$\alpha_{tv} + \beta_{tv} \leq 1, \quad \forall \, t \in T, \, o \in O, \, v \in L_\omega$$ (48)

We remark that a subset of feasible solutions may not satisfy Theorem 5.1; therefore, the inequalities it yields might not be strictly valid inequalities. Furthermore, Theorem 5.1 can induce different constraints other than (48) and it is still unclear whether (48) is the most proper formulation.
Corollary 5.2 can be obtained as a direct consequence of Theorem 5.1, and thus, it suffices to prove Theorem 5.1. For a single-block warehouse, let the artificial locations in the first cross-aisle be $u_1, u_2, \ldots, u_n$, and the artificial locations in the second cross-aisle be $d_1, d_2, \ldots, d_n$. Let picking locations in the path north to south in aisle $i$ be $v_{i1}, v_{i2}, \ldots, v_{il_i}$. We illustrate all of these concepts in Figure 6. A Eulerian tour can be specified using a sequence of vertices, such as

$$r = (u_1v_{11}v_{12} \ldots d_1d_2 \ldots u_2u_1)$$

Let $V(r)$ denotes the set of the vertices in $r$. We say that Eulerian tour $r'$ is better than Eulerian tour $r$ if $r'$ satisfy

1. $\{V(r) \cap V_L\} \subseteq \{V(r') \cap V_L\}$
2. $r'$ is shorter than $r$

Now, we introduce three lemmas to prove Theorem 5.1.

Lemma 5.3: For any tour $r$, if there exists an aisle $i$ such that tour $r$ contains subpath $(u_{i-1}u_i v_{i1}v_{i2}v_{i3}d_i d_{i+1} \ldots v_{ik} v_{k1} v_{k2} \ldots v_{kl} v_{l1} u_i)$, then there exists a tour $r'$ better than $r$.

Proof: We simply assume that $r = (r_1r_2 \ldots r_p u_i v_{i1}v_{i2} \ldots v_{ik} d_i d_{i+1} \ldots v_{kl} v_{l1} u_i \ldots r_R)$. Then $r' = (r_1r_2 \ldots r_p u_i v_{i1}v_{i2} \ldots v_{ik} d_i d_{i+1} \ldots v_{kl} v_{l1} u_i \ldots v_{km} v_{ml} d_m d_{m+1} \ldots v_{kn} v_{ln} u_i \ldots r_R)$ is a better tour.

The reader may find a different application of Lemma 5.3 in Valle, Beasley, and Salles da Cunha (2017), in which it also induced the artificial vertex reversal cuts.

Lemma 5.4: For any tour $r$, if there exists an aisle $i$ such that tour $r$ contains subpath $(u_{i-1}u_i v_{i1}v_{i2} \ldots v_{ik} d_i d_{i+1})$ and $(d_i v_{ik} v_{ik+1} \ldots v_{kl} v_{l1} u_i)$, then there exists a tour $r'$ better than $r$.

Proof: We simply assume that $r = (r_1r_2 \ldots r_p u_i v_{i1}v_{i2} \ldots v_{ik} d_i d_{i+1})$. Then $r' = (r_1r_2 \ldots r_p u_i v_{i1}v_{i2} \ldots v_{ik} d_i d_{i+1} \ldots v_{kl} v_{l1} u_i \ldots r_R)$ is a better tour.

Now, we can give the proof for Theorem 5.1.

Proof: We assume by contradiction that there exists an optimal tour $r = (r_1r_2 \ldots r_p u_{i-1}u_i v_{i1}v_{i2} \ldots v_{ik} d_i d_{i+1} \ldots v_{kl} v_{l1} u_i \ldots r_R)$ and $r$ contains subpath $(d_i v_{ik} v_{ik+1} \ldots v_{kl} v_{l1} u_i)$. Then

1. If $r_q r_{q+1} \ldots r_{q+k} = (v_{ik} v_{ik+1} \ldots v_{kl} u_i)$, Lemma 5.3 provides a better tour.
2. If $r_q = d_{i+1}$, Lemma 5.4 provides a better tour.
3. If $r_q = d_{i-1}$, Lemma 5.5 provides a better tour.

Thus, no such optimal tour exists.

In the remainder of this subsection, we extend the result for a 2-block warehouse. Let $l$ be the index of the topmost left subaisle, and we have the following theorem.

Theorem 5.6: For a 2-block warehouse, there exists an optimal walk that traverses any subaisle $i \in \{2, 3, \ldots, \}$ at most once.

Proof: Let $r$ be an optimal walk. We can partition this tour into several parts $r_1, r_2, \ldots, r_R, r_0$ and each part
is located in the first block or the second block (see Figure 7). As shown in the proof of Theorem 5.1, a subaisle cannot be traversed twice by any subtour \( r_k, k \in \{0, 1, \ldots, R\} \). Furthermore, there always exists an optimal solution such that a subaisle will not be traversed by different subtours \( r_k, r_t \), where \( k, t \in \{1, 2, \ldots, R\} \). The construction of such an optimal solution is similar to the proofs of Lemma 5.4-5.5. Thus, there exists an optimal walk \( r \) such that any subaisle is traversed at most once by \( r' = (r_1, r_2, \ldots, r_k) \). If a picker needs to pick some products in \( r_0 \), then any subaisle is traversed at most once by \( r \); otherwise, the picker needs to return to the origin immediately, and we can assume that the picker first goes to the south artificial vertex of subaisle 1 and then goes to the origin. This completes the proof. 

**Corollary 5.7 (Single traversing constraints):** For a 2-block warehouse, there exists an optimal solution satisfying the following constraints:

\[
\alpha_{tw} + \beta_{tv} \leq 1, \quad \forall \ t \in T, \ o \in O, \ v \in L_0 \setminus V_{sub}(1)
\]  

**Proof:** This follows immediately from Theorem 5.6. ■

**6. No-reversal case**

In this section, we introduce the no-reversal case of the JOBPRP and present new formulations that are less influenced by symmetric solutions. We note that it would be impractical to search for an optimal solution when there are a vast number of orders. By assuming that routing is conducted in a no-reversal fashion, Valle, Beasley, and Salles da Cunha (2017) obtained an efficiently solvable formulation. The no-reversal constraints are described as follows.

\[
x_{int(v)\nu} = x_{tv}, \quad \forall \ t \in T, \ i \in [W_{sub}], \ \nu \in V_{sub}(i) \cup \{l(i)\}
\]  

In the case of no-reversal constraints, a picker is not allowed to reverse in a subaisle. Many feasible solutions, including some or all optimal solutions, will be cut off by these constraints. Note that the no-reversal JOBPRP can also be regarded as an approximate problem, and by solving this problem, we can quickly make order batching decisions. Then each picker can be rerouted by applying other methods such as a classical TSP model.

Although these constraints can greatly simplify the decision problem, this problem still suffers severely from the presence of symmetry. A large amount of equivalence of solutions induced by symmetry routes might confound the branch-and-bound (or branch-and-cut) process, as illustrated in Figure 8. In other words, there is also a substantial room for the further improvement of the no-reversal formulation.

**6.1. TSP formulation for a single-block warehouse**

A single-block warehouse is first considered. We start with an undirected graph \( G^+ = (V_1, E^+) \), which is constructed as follows: Recall that \( e = [u, v] \) denote the unordered pair of location \( u \) and location \( v \). \( E^+ \) is defined as \( E_1^+ \cup E_2^+ \), where \( E_1^+ = \{[u, v] : (u, v) \in E' \} \) and \( E_2^+ = \{[s, v] : v \in V_1 \setminus \{s\} \} \). For any subaisle \( i \), let \( e(i) \) denote the edge \([f(i), l(i)]\). The resulting feasible region \( P_1^U \) then consists of the following constraints.

\[
x_{[s(l(1))]f(1)} + x_{[s(f(2))]l(1)} \geq 1, \quad \forall \ t \in T
\]  

\[
\sum_{[s,v] \in \delta(s)} x_{t[s,v]} + \tilde{x}_t = 2, \quad \forall \ t \in T
\]  

\[
x_{t[u,v]} \geq z_{ot}, \quad \forall \ t \in T, \ i \in [W_{sub}], \ [u,v] = e(i), \ o \in \{o : V_{sub}(i) \cap L_o \neq \emptyset\}
\]  

\[
\sum_{[l(i),v] \in \delta(l(i))} x_{t[l(i),v]} + \tilde{x}_t = 2y_{lu}, \quad \forall \ t \in T
\]  

\[
\sum_{[u,v] \in \delta(u)} x_{t[u,v]} = 2y_{lu}, \quad \forall \ t \in T, \ u \in V_1 \setminus \{s,l(1)\}
\]  

\[
\sum_{[u,v] \in \delta(S)} x_{t[u,v]} \geq 2y_{lu_0}, \quad \forall \ t \in T, \ S \subset V_1 \setminus \{s\}, |S| \geq 2,
\]

\[
u_0 \in S
\]  

\[
z_{ot} = 1, \quad \forall \ o \in O
\]  

\[
\sum_{o \in O} b_{o}z_{ot} \leq B, \quad \forall \ t \in T
\]  

\[
x_{t[u,v]} \in \{0, 1\}, \quad \forall \ t \in T, \ [u,v] \in E^+
\]

\[
\tilde{x}_t \in \{0, 1\}, \quad \forall \ t \in T
\]  

\[
y_{kv} \in \{0, 1\}, \quad \forall \ t \in T, \ v \in V_1
\]  

\[
z_{ot} \in \{0, 1\}, \quad \forall \ t \in T, \ o \in O
\]  

Here, \( x_{t[u,v]} \) is a binary variable, which takes the value 1 if and only if edge \([u, v]\) is traversed by picker \( t \). We let binary variable \( \tilde{x} \) indicate whether a parallel edge between \( f(1) \) and \( s(1) \) should be added (note that the origin \( s = f(1) \)). We actually formulate the standard TSP on \( E^+ \) (possibly with a parallel edge of \( e(1) \)) using constraints (52)–(57). We illustrate how our formulation works in Figure 9.

The TSP-based formulation is highly dependent on the construction of the auxiliary graph. Once we define the undirected graphs for other warehouses, we are able
The picker may require to traverse a subaisle twice in the block warehouse such that:

1. The picker may require to traverse a subaisle twice to enter the other block.
2. The picker may require to traverse some edge in the second cross-aisle twice before retrieving all products from storage (after which the picker should go back to the origin).

From the above observations, we should add more parallel edges to the graph to preserve optimal solutions. In fact, we are describing the problem property when adding parallel edges. Once the auxiliary graph maintains a subset of optimal solutions, we obtain a TSP-based formulation.

Now, we show how to construct the auxiliary graph. Let $W$ be a set of artificial locations in the second cross-aisle. We construct a copy $v'$ for any $v \in W$ and let the set of copies be $W'$. Let $E_1^+$ be the set of edges connecting neighboring vertex. $E_2^+$ and $E_3^+$ are defined as follows.

1. $E_1^+ = \{ [v', Q_s(v')] : v \in W \} \cup \{ [v, Q_s(v')] : v \in W \}$
2. $E_2^+ = \{ [s, v] : v \in \{ V \cup W' \} \setminus \{ s \} \}$
3. $E_3^+$ enables the picker to return to the origin after picking is completed. We compare the two auxiliary graphs in this section, as shown in Figure 10.

For each picker $t \in T$, let $x_{[u,v]} \in \{0,1\}$ be an indicator variable equal to 1 if $[u,v] \in E_1^+ \cup E_2^+$ is traversed, and $\tilde{x}_{[u,v]} \in \{0,1\}$ be an indicator variable equal to 1 if $[u,v] \in E_3^+$ is traversed. The TSP-based formulation $P_U^2$ is similar to $P_U^1$, and is described by the following constraints.

\[
\sum_{[s,v] \in \delta(s)} x_{[s,v]} \geq 1, \quad \forall t \in T \tag{64}
\]

\[
\sum_{[s,v] \in \delta(s)} (x_{[s,v]} + \tilde{x}_{[s,v]}) = 2, \quad \forall t \in T \tag{65}
\]

\[
x_{[u,v]} \geq z_{ot}, \quad \forall t \in T, i \in W_{sub}, [u,v] = e(i), o \in \{ o : V_{sub}(i) \cap L_o \neq \emptyset \} \tag{66}
\]

\[
\sum_{[u,v] \in \delta(u)} (x_{[u,v]} + \tilde{x}_{[u,v]}) = 2y_{tu_0}, \quad \forall t \in T, u \in V_I \setminus \{ s \} \tag{67}
\]

\[
\sum_{t \in T} z_{ot} = 1, \quad \forall o \in O \tag{69}
\]

\[
\sum_{o \in O} b_{o} z_{ot} \leq B, \quad \forall t \in T \tag{70}
\]

\[
x_{[u,v]} \in \{0,1\}, \quad \forall t \in T, [u,v] \in E_1^+ \cup E_2^+ \tag{71}
\]

\[
\tilde{x}_{[u,v]} \in \{0,1\}, \quad \forall t \in T, [u,v] \in E_3^+ \tag{72}
\]

\[
y_{tv} \in \{0,1\}, \quad \forall t \in T, v \in V_I \tag{73}
\]
The existence of such a tour.

Theorem 6.1: There always exists an optimal picking tour which is done in an S-shape fashion.

Proof: We first consider the case where subaisle \( 1 \in K_1 \). Let the total vertical distance of route \( r \) be \( l_r \) and the optimal route \( r^* \). Let \( d \) denote the length of a subaisle. It can be seen that \( l_{r^*} \geq |K_1 \cup K_2|d \). We also have the following results for \( r^*_1 \):

1. If \( |K_1| \) and \( |K_2| \) are odd, we have that \( l_{r^*_1} - |K_1 \cup K_2|d = 2d \) when \( 1 \in K_1 \).
2. If \( |K_1| \) and \( |K_2| \) are even, we have that \( l_{r^*_1} - |K_1 \cup K_2|d = 0 \) when \( 1 \in K_1 \).
3. If \( |K_1| \) is odd and \( |K_2| \) is even, we have that \( l_{r^*_1} - |K_1 \cup K_2|d = |1 \in K_1 \).
4. If \( |K_1| \) is even and \( |K_2| \) is odd, we have that \( l_{r^*_1} - |K_1 \cup K_2|d = d \) when \( 1 \in K_1 \).

We now show that \( r^*_1 \) is an optimal picking tour. We only deal with the case when \( |K_1| \) is odd and \( |K_2| \) is even. Assume by contradiction that \( l_{r^*} = |K_1 \cup K_2|d \), which means \( r^* \) will traverse each selected subaisle exactly once. \( r^* \) can be described as a sequence of subaisles \( i_1, i_2, \ldots, i_n \). If \( i_1, i_k \in K_1 \) and \( i_2, \ldots, i_{k-1} \in K_2 \), we can find that \( k \) is even. \( i_1, i_2, \ldots, i_n \) can be reformulated as \( Q_1 = \{i_1, i_2, \ldots, i_{k_1}\}, Q_2 = \{i_{k_1+1}, i_2, \ldots, i_{k_2}\}, \ldots, Q_{2n+1} = \{i_{2n+1}, i_{2n+2}, \ldots, i_{2n+1}\} \) where \( Q_{2t+1} \subset K_1 \) and \( Q_{2t} \subset K_2 \). Note that \( |Q_2|, \ldots, |Q_{2n}| \) are even and \( |Q_1| \) is odd, we can assume that \( |Q_1| \) is even. This implies that \( r^* \) must traverse \( i_{k_1} \) twice, which is a contradiction.

\[ z_{ot} \in \{0, 1\}, \quad \forall t \in T, o \in O \]
Similarly, we can prove that there exist an optimal picking tour which is of type $r_1^1$ or $r_2^1$ when $1 \notin K_1$. This completes the proof.

Note that the second cross-aisle is passed through by $r_1^2$ or $r_2^2$ exactly twice, and thus, we can further tighten the feasible region. Let $V_S$ be the set of vertices in the second block, i.e. $V_S = \{W' \cup \{Q_S(v) : v \in W'\}\}$. We use the following constraint to ensure that the second cross-aisle can be passed through at most twice by the picker. This constraint is very effective at reducing solution times.

$$\sum_{[u,v] \in \delta(V_S)} (x_{t[u,v]} + \bar{x}_{t[u,v]}) \leq 2, \quad \forall t \in T$$ (75)

We finish this section with the following corollary.

**Corollary 6.2:** There always exists an optimal solution satisfying constraints (75).

**Proof:** This follows immediately from Theorem 6.1. $\blacksquare$

## 7. Computational results

The experiments were performed on an AMD Ryzen 7 4800H @2.90 GHz processor and 32 GB of RAM. The code was written in Python and GUROBI 9.1.1 was used as the mixed-integer solver. The performances of our MIP formulations and additional constraints are tested by comparing the computational difficulties to find optimal solutions.

### 7.1. Test problems

Our formulations are tested over the publicly available benchmark instances at [http://www.dcc.ufmg.br/arbex/orderpicking.html](http://www.dcc.ufmg.br/arbex/orderpicking.html). It comes from a database of anonymized customer purchases over two years for a chain of supermarkets. There are a total of 1560 distinct products in the database. A single order is generated by combining the purchases of a customer over the first $\Delta$ days with $\Delta \in \{5, 10, 20\}$. The number of orders varies from 5 to 30. The warehouse layouts considered in our experiments are similar to that of Valle, Beasley, and Salles da Cunha (2017). Both single-block and 2-block warehouses contain 8 aisles, each of which contains 33 slots.
Table 2. Comparison of the branch-and-cut algorithm based on formulations $P_O$, $P_G^+$ and $P_F^+$.

| $\Delta$ | O | $T$(seconds) | UB | LB | GAP(%) | $T$(seconds) | UB | LB | GAP(%) | $T$(seconds) | UB | LB | GAP(%) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | 5 | 0.2 | 346 | 346 | 0 | 0.24 | 346 | 346 | 0 | 0.51 | 346 | 346 | 0 |
| 10 | 2.91 | 578 | 578 | 0 | 1.5 | 578 | 578 | 0 | 236 | 578 | 578 | 0 |
| 15 | 15 | 650 | 650 | 0 | 7.8 | 650 | 650 | 0 | 691 | 650 | 650 | 0 |
| 16 | 390 | 766 | 766 | 0 | 37 | 766 | 766 | 0 | 361 | 766 | 766 | 0 |
| 17 | 90 | 802 | 802 | 0 | 81 | 802 | 802 | 0 | 690 | 802 | 802 | 0 |
| 18 | 1821 | 766 | 766 | 0 | 37 | 766 | 766 | 0 | 236 | 766 | 766 | 0 |
| 19 | 90 | 802 | 802 | 0 | 81 | 802 | 802 | 0 | 690 | 802 | 802 | 0 |
| 20 | 1821 | 766 | 766 | 0 | 37 | 766 | 766 | 0 | 236 | 766 | 766 | 0 |
| 21 | 1104 | 815 | 815 | 0 | 81 | 815 | 815 | 0 | 236 | 815 | 815 | 0 |
| 22 | 2400 | 856 | 856 | 0 | 136 | 856 | 856 | 0 | 236 | 856 | 856 | 0 |
| 23 | 2400 | 856 | 856 | 0 | 136 | 856 | 856 | 0 | 236 | 856 | 856 | 0 |
| 24 | 2400 | 856 | 856 | 0 | 136 | 856 | 856 | 0 | 236 | 856 | 856 | 0 |
| 25 | 2400 | 856 | 856 | 0 | 136 | 856 | 856 | 0 | 236 | 856 | 856 | 0 |

Note: The symbol ‘–’ shows that GUROBI failed to find a feasible solution.

7.2. Computational results

In this section, we compare our formulations with that of Valle, Beasley, and Salles da Cunha (2017). The formulation for the JOBPRP (resp., for the no-reversal JOBPRP) presented by Valle, Beasley, and Salles da Cunha (2017) is denoted as $P_O$ (resp., $P_U$). To further improve our formulation, we take into account existing constraints including aisles cuts, artificial vertex reversal constraints (Valle, Beasley, and Salles da Cunha 2017) and column inequalities (Kaibel and Pfetsch 2006). To have a better comparison, we also use column inequalities to improved $P_O$. We

on each side. The single-block warehouse contains one Northern cross-aisle and one Southern cross-aisle while the 2-block warehouse contains an extra cross-aisle with 17 slots in the Northern block and 16 slots in the Southern block. The aisle and cross-aisle widths are 3 metres, and both the slot width and rack depth are 1 metre. The 2-block warehouse layout is used for the majority of the work reported here and the single-block warehouse layout is only used for the test of the formulations $P_U$ and $P_{U^+}$. It should be noted that we assume the origin is right from the first artificial location while (Valle, Beasley, and Salles da Cunha 2017) set the distance from the origin to the first artificial vertex is 4 meters. However, our formulations for their warehouse layout are essentially the same. We set the available capacity of the picking vehicle $B = 8$. For every test instance, we define $T$ by solving a bin-packing problem.
do not present experimental results to verify the effectiveness of these constraints that existing studies have already illustrated. Details regarding test formulations are shown in Table 1.

Remark that all formulations except $P_F^+$ are non-compact. The exponentially many constraints are generated sequentially in a branch-and-cut framework. For each candidate integral solution, we add constraints when the graph for some picker is disconnected. The connectivity condition is verified by a depth-first search. In addition, $P_F$ seems unable to benefit from aisles cuts and therefore $P_F^+$ does not include aisles cuts.

In Table 2, we compare formulation $P_O$, $P_G^+$ and $P_F^+$ on the selected instances by setting a time limit of 2400 s. Column $O$ corresponds to the number of orders. Column $T$ corresponds to the total time. Columns $UB$ and $LB$ represent the best upper and lower bounds obtained at the end of the search, respectively, when either the instance was solved to prove optimality or the time limit has hit. GAP is defined as $100\% \times \frac{UB-LB}{UB}$.

The branch-and-cut algorithm based on $P_G^+$ managed to solve most instances to proven optimality. Furthermore, it obtained the lowest gap or had the shortest computing time. Thus, we can state that $P_G^+$ outperforms the existing formulation $P_O$. However, $P_F^+$ is not as strong as $P_G^+$. Although it provided a better gap (than $P_O$) for some instances (for example, instance with $\Delta = 5, O \in \{19, 30\}$), it performs poorly when $\Delta = 10$. Furthermore, an increasing number of orders cause a fast increase in the solution time, even for $P_G^+$. One possible reason is that as the number of order pickers increases, the number of symmetry branches in the search tree grows exponentially. In fact, even the relaxation $\min \{f(x) : (x, z, \alpha, \beta) \in P_{sub}, \text{constraints} \ (9) - (10), \ (13) - (16), \ (18)\}$ (see Figure 3(c)) is very difficult to solve when there are a large number of orders (and it cannot figure out the route for each picker).

We analyze the efficiency of the basic cuts and the single traversing constraints by adding them to $P_O$, $P_G^+$ and $P_F^+$. Note that the basic cuts are generated by performing a depth-first search algorithm, and the total running time is typically less than 0.5 seconds (0.03 s – 0.5 s). Therefore we do not need to take into account the processing time of constructing the basic cuts. The original model and the strengthened model are compared by counting winning instances. An instance is a winner for model A compared with model B, if

1. $A$ finished within the time limit and $B$ did not finish or required a longer CPU time or
2. $A$ obtained a lower gap than $B$.

| $\Delta$ | $P_O$ Win (%) | $P_O + \text{basic cuts}$ Win (%) |
|---------|---------------|----------------------------------|
| 5       | 69.2          | 30.8                             |
| 10      | 46.2          | 53.8                             |
| 20      | 30.8          | 69.2                             |

| $\Delta$ | $P_G^+$ Win (%) | $P_G^+ + \text{basic cuts}$ Win (%) |
|---------|----------------|----------------------------------|
| 5       | 25             | 75                               |
| 10      | 41.7           | 58.3                             |
| 20      | 46.2           | 53.8                             |

| $\Delta$ | $P_F^+$ Win (%) | $P_F^+ + \text{basic cuts}$ Win (%) |
|---------|----------------|----------------------------------|
| 5       | 58.3           | 41.7                             |
| 10      | 53.8           | 46.2                             |
| 20      | 71.4           | 28.6                             |

If the difference between the times or gaps is below 1 s or 0.1%, respectively, the instance is not counted. For example, we can compare $P_O$ and $P_F^+$ in Table 2 by only considering the instances with $\Delta = 5$. Then we can observe that $P_O$ has 3 winners and $P_F^+$ has 10 winners. Tables 3 and 4 show the impacts of adding these additional constraints. Except the aggregated results, we also provide more detailed results in the appendix.

Table 3 shows the efficiency of the basic cuts. For formulation $P_O$, the basic cuts can improve at most 69.2% of the instances (when $\Delta = 20$). Similarly, the basic cuts can improve at most 75% of the instances for $P_G$ (when $\Delta = 5$). However, due to being compact and not requiring an explicit branch-and-cut implementation, $P_F^+$ seems to benefit less from the basic cuts. Table 4 shows the efficiency of the single traversing constraints. Although both $P_G^+$ (at most 75% of the instances) and $P_F^+$ (at most
Table 5. Comparison of the branch-and-cut algorithm based on formulations $P_U$ and $P^{1+}_U$ for a single-block warehouse.

| $\Delta$ | O | T(seconds) | UB | LB | GAP(%) | T(seconds) | UB | LB | GAP(%) |
|---|---|---|---|---|---|---|---|---|---|
| 5  | 5  | 0.02 | 358 | 358 | 0  | 0.01 | 358 | 358 | 0  |
| 10 | 0.13 | 634 | 634 | 0  | 0.05 | 634 | 634 | 0  |
| 15 | 0.05 | 716 | 716 | 0  | 0.01 | 716 | 716 | 0  |
| 20 | 1.72 | 982 | 982 | 0  | 0.16 | 982 | 982 | 0  |
| 21 | 26  | 1064 | 1064 | 0  | 0.21 | 1064 | 1064 | 0  |
| 22 | 41  | 1064 | 1064 | 0  | 0.25 | 1064 | 1064 | 0  |
| 23 | 20  | 1064 | 1064 | 0  | 0.27 | 1064 | 1064 | 0  |
| 24 | 300 | 1248 | 1140 | 8.7 | 1.64 | 1248 | 1248 | 0  |
| 25 | 98  | 1258 | 1258 | 0  | 2.21 | 1258 | 1258 | 0  |
| 26 | 64  | 1268 | 1268 | 0  | 1.3  | 1268 | 1268 | 0  |
| 27 | 215 | 1278 | 1278 | 0  | 1.17 | 1278 | 1278 | 0  |
| 28 | 235 | 1330 | 1330 | 0  | 2.62 | 1330 | 1330 | 0  |
| 29 | 300 | 1350 | 1340 | 0.7 | 1.2  | 1350 | 1350 | 0  |
| 30 | 20  | 1064 | 1064 | 0  | 1.66 | 1350 | 1350 | 0  |

76.9 of the instances) are able to benefit from the single traversing constraints, more than half of instances $P_O$ cannot be improved by these constraints. We also note that the basic cuts and the single traversing constraints sometimes increase the solution time, which could be due to the interaction of these constraints and some built-in general-purpose cuts. Furthermore, we believe that the single traversing constraints should be given more consideration; This can lead to a much smaller feasible region and may induce other constraints or formulations.

In Table 5, we compare formulation $P_U$ and $P^{1+}_U$ for a single-block warehouse setting a time limit of 300 s. We reduce the time limit, mainly because the no-reversal JOBPRP is much simpler than the JOBPRP. The branch-and-cut algorithm based on $P^{1+}_U$ can solve all instances within several seconds. The main reason could be that we successfully cut many symmetric solutions by designing an auxiliary graph. Similarly, we compare formulation $P_U$ and $P^{2+}_U$ for a 2-block warehouse in Table 6, and $P^{2+}_U$ still outperforms $P_U$ for all instances.

8. Conclusions

In this article, we investigate the JOBPRP, which is pivotal for the efficiency of order picking operations. To fully utilize the structure of the warehouse, we reconstruct the connectivity constraints. The obtained formulations consider separately the graph properties of picking locations and artificial locations. In our experiments, much less computation time is required to solve the improved formulations, which means we successfully enhance the original formulation by reducing the difficulty caused by the routing problem. We also provide two types of relevant additional constraints: one aims at dealing with
batching decisions and routing decisions in an integrated way; the other aims at cutting off a subset of feasible solutions by the property of an optimal routing. Sometimes they improve computational results and sometimes they lead to worse results. These constraints may be further improved by a polyhedral study for the warehouse layout. Additionally, we consider the optimal routing for the no-reversal special case of this problem, which can be easier to implement in practice. Our experimental results show that the no-reversal formulations give a good trade-off between computation time and solution quality. Furthermore, we can focus on a specific type of picking tour by embedding additional constraints into a formulation, which helps to achieve computational efficiency.

There are several potential topics for future research. First, graph-based mathematical formulations should consider the warehouse structure, which implies a need for polyhedral studies of different warehouses. For example, one might investigate the graph representation and the associated polytope for the HappyChic warehouse considered by Briant et al. (2020), which is slightly different from the rectangular warehouse considered in this paper. Second, one might improve traditional heuristic algorithms by analyzing the property of optimal solutions. Third, both the routing and batching problems suffer severely from the presence of symmetry. If we treat the batching problem as a partitioning problem, we can find many symmetry breaking methods (for example, column inequalities Kaibel and Pfetsch 2006). One might make use of these symmetry breaking methods to improve different heuristics or exact methods. Fourth, as no-reversal routes are easy to implement in practice, it might be worthwhile to pay more attention to this special case. Arbex Valle and Beasley (2020) demonstrated

| | | PU | | | | P2+U | |
|---|---|---|---|---|---|---|---|
| | | T(seconds) | UB | LB | GAP(%) | T(seconds) | UB | LB | GAP(%) |
| 5 | 5 | 0.05 | 382 | 382 | 0 | 0.02 | 382 | 382 | 0 |
| 10 | 0.51 | 608 | 608 | 0 | 0.2 | 608 | 608 | 0 |
| 15 | 1.47 | 696 | 696 | 0 | 0.22 | 696 | 696 | 0 |
| 20 | 20 | 940 | 940 | 0 | 1.91 | 940 | 940 | 0 |
| 21 | 5.34 | 940 | 940 | 0 | 1.73 | 940 | 940 | 0 |
| 22 | 7.18 | 940 | 940 | 0 | 1.2 | 940 | 940 | 0 |
| 23 | 8.2 | 950 | 950 | 0 | 1.54 | 950 | 950 | 0 |
| 24 | 273 | 1108 | 1108 | 0 | 31 | 1108 | 1108 | 0 |
| 25 | 152 | 1146 | 1146 | 0 | 41 | 1146 | 1146 | 0 |
| 26 | 300 | 1194 | 1118 | 6.4 | 45 | 1176 | 1176 | 0 |
| 27 | 300 | 1206 | 1185 | 1.7 | 52 | 1206 | 1206 | 0 |
| 28 | 300 | 1254 | 1151 | 6.7 | 67 | 1254 | 1254 | 0 |
| 29 | 300 | 1234 | 1184 | 5.6 | 67 | 1254 | 1254 | 0 |
| 30 | 300 | 1254 | 1151 | 8.2 | 48 | 1254 | 1254 | 0 |
| 10 | 5 | 0.02 | 382 | 382 | 0 | 0.02 | 382 | 382 | 0 |
| 10 | 1.29 | 724 | 724 | 0 | 0.29 | 724 | 724 | 0 |
| 15 | 4.74 | 922 | 922 | 0 | 3.08 | 922 | 922 | 0 |
| 20 | 8 | 1020 | 1020 | 0 | 0.97 | 1020 | 1020 | 0 |
| 21 | 132 | 1058 | 1058 | 0 | 1.42 | 1058 | 1058 | 0 |
| 22 | 26 | 1058 | 1058 | 0 | 1.13 | 1058 | 1058 | 0 |
| 23 | 237 | 1214 | 1214 | 0 | 19 | 1254 | 1254 | 0 |
| 24 | 266 | 1254 | 1254 | 0 | 14 | 1254 | 1254 | 0 |
| 25 | 213 | 1321 | 1302 | 1.5 | 14 | 1321 | 1302 | 0 |
| 26 | 300 | 1321 | 1302 | 1.5 | 17 | 1321 | 1302 | 0 |
| 27 | 300 | 1342 | 1302 | 3 | 35 | 1342 | 1342 | 0 |
| 28 | 300 | 1352 | 1231 | 8.9 | 23 | 1352 | 1352 | 0 |
| 29 | 300 | 1352 | 1264 | 6.5 | 20 | 1352 | 1352 | 0 |
| 30 | 300 | 1352 | 1255 | 7.2 | 22 | 1352 | 1352 | 0 |
| 20 | 5 | 0.2 | 620 | 620 | 0 | 0.09 | 620 | 620 | 0 |
| 10 | 2.68 | 982 | 982 | 0 | 0.69 | 982 | 982 | 0 |
| 15 | 59 | 1108 | 1108 | 0 | 0.67 | 1108 | 1108 | 0 |
| 20 | 300 | 1430 | 1411 | 1.5 | 11 | 1430 | 1430 | 0 |
| 21 | 300 | 1638 | 1513 | 7.6 | 154 | 1626 | 1626 | 0 |
| 22 | 300 | 1646 | 1526 | 7.3 | 114 | 1626 | 1626 | 0 |
| 23 | 300 | 1684 | 1534 | 8.9 | 162 | 1648 | 1648 | 0 |
| 24 | 300 | 1744 | 1600 | 8.2 | 234 | 1708 | 1708 | 0 |
| 25 | 300 | 1718 | 1565 | 8.9 | 125 | 1714 | 1714 | 0 |
| 26 | 300 | 1746 | 1562 | 10.5 | 148 | 1716 | 1716 | 0 |
| 27 | 300 | 1756 | 1589 | 9.5 | 179 | 1736 | 1736 | 0 |
| 28 | 300 | 1932 | 1430 | 26 | 300 | 1894 | 1629 | 18.8 |
| 29 | 300 | 1990 | 1444 | 27.4 | 300 | 1932 | 1629 | 15.7 |
| 30 | 300 | 2066 | 1476 | 28.6 | 300 | 1990 | 1734 | 12.9 |
the feasibility of using easy-to-solve approximation programs to obtain high-quality no-reversal solutions. One might build up an approximation model that only considers some features of a feasible solution, and might study the accuracy of the estimation.

Acknowledgements
The authors thank three anonymous referees for helpful comments that led to significant improvements of the paper.

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

Funding
Thanks are also given to the financial support from the National Nature Science Foundation of China under Grant No. 12071428 and 6211130247, and the Zhejiang Provincial Natural Science Foundation of China under Grant No. LZ20A010002.

Notes on contributors
Kai Zhang is a Ph.D. student majoring in Operational Research and Cybernetics at Zhejiang University. He received a bachelor’s degree in applied mathematics from the School of Mathematical Sciences, Zhejiang University, Hangzhou, China in 2018. His current research interests primarily include operations research, supply chain management, 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 Postdoctoral Researcher 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 optimization, 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.

Data availability statement
The data that support the findings of this study are available from the corresponding author, C.H. Gao, upon reasonable request.

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

References
Aerts, Babiche, Trijntje Cornelissens, and Kenneth Sörensen. 2021. “The Joint Order Batching and Picker Routing Problem: Modelled and Solved As a Clustered Vehicle Routing Problem.” Computers & Operations Research 129: 105168.

André, Scholz, Daniel Schubert, and Gerhard Wäscher. 2017. “Order Picking with Multiple Pickers and Due Dates – Simultaneous Solution of Order Batching, Batch Assignment and Sequencing, and Picker Routing Problems.” European Journal of Operational Research 263 (2): 461–478.

Attari, Mahdi Yousefi Nejad, Ali Ebadi Torkayesh, Behnam Malmir, and Ensiyeh Neyshabouri Jami. 2021. “Robust Possibilistic Programming for Joint Order Batching and Picker Routing Problem in Warehouse Management.” International Journal of Production Research 59 (14): 4434–4452.

Briant, Olivier, Hadrien Cambazard, Diego Cattaruzza, Nicolas Catusse, Anne-Laure Ladier, and Maxime Ogier. 2020. “An Efficient and General Approach for the Joint Order Batching and Picker Routing Problem.” European Journal of Operational Research 285 (2): 497–512.

Cambazard, Hadrien, and Nicolas Catusse. 2018. “Fixed-parameter Algorithms for Rectilinear Steiner Tree and Rectilinear Traveling Salesman Problem in the Plane.” European Journal of Operational Research 270 (2): 419–429.

Cergibozan, Çağla, and A. Tosan. 2019. “Order Batching Operations: An Overview of Classification, Solution Techniques, and Future Research.” Journal of Intelligent Manufacturing 30: 335–349.

Chen, Tzu-Li, Chen-Yang Cheng, Yin-Yann Chen, and Li-Kai Chan. 2015. “An Efficient Hybrid Algorithm for Integrated Order Batching, Sequencing and Routing Problem.” International Journal of Production Economics 159: 158–167.

Cheng, Chen-Yang, Yin-Yann Chen, Tzu-Li Chen, and John Jung-Woon Yoo. 2015. “Using a Hybrid Approach Based on the Particle Swarm Optimization and Ant Colony Optimization to Solve a Joint Order Batching and Picker Routing Problem.” International Journal of Production Economics 170: 805–814. Decision models for the design, optimization and management of warehousing and material handling systems, https://www.sciencedirect.com/science/article/pii/S0925527315000894.

Cornuéjols, Gérard, Jean Fonlupt, and Denis Naddef. 1985. “The Traveling Salesman Problem on a Graph and Some Related Integer Polyhedra.” Mathematical Programming 33: 1–27.

Defryn, Christof, and Kenneth Sörensen. 2017. “A Fast Two-level Variable Neighborhood Search for the Clustered Vehicle Routing Problem.” Computers & Operations Research 83: 78–94.
de Koster, René, Tho Le-Duc, and Kees Jan Roodbergen. 2007. “Design and Control of Warehouse Order Picking: A Literature Review.” *European Journal of Operational Research* 182 (2): 481–501.

Elsayed, E. A. 1981. “Algorithms for Optimal Material Handling in Automatic Warehousing Systems.” *International Journal of Production Research* 19 (5): 525–535.

Gademann, A.J.R.M. (NOUD), Jeroen P. van den Berg, and Hassan H. van der Hoff. 2001. “An Order Batching Algorithm for Wave Picking in a Parallel-aisle Warehouse.” *IEEE Transactions* 33 (5): 385–398.

Goetschalckx, Marc, and H. Donald Ratliff. 1988. “Order Picking In An Aisle.” *IEEE Transactions* 20 (1): 53–62.

Grosse, E. H., C. H. Glock, and R. Ballester-Ripoll. 2014. *A simulated annealing approach for the joint order batching and order picker routing problem with weight restrictions*. Publications of Darmstadt Technical University, Institute for Business Studies (BWL) 65331. Darmstadt Technical University, Department of Business Administration, Economics and Law, Institute for Business Studies (BWL). https://ideas.repec.org/p/dar/wpaper/65331.html.

Hall, Randolph W. 1993. “Distance Approximations for Routing Manual Pickers in a Warehouse.” *IEEE Transactions* 25 (4): 76–87.

Helsgaun, Keld. 2000. “An Effective Implementation of the Lin−Kernighan Traveling Salesman Heuristic.” *European Journal of Operational Research* 126 (1): 106–130.

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: 3391–3417.

Hong, Soon do, Andrew L. Johnson, and Brett A. Peters. 2012. “Large-scale Order Batching in Parallel-aisle Picking Systems.” *IEEE Transactions* 44 (2): 88–106.

Kaibel, Volker, and Marc Pfetsch. 2006. “Packing and Partitioning Orbitopes.” *Mathematical Programming* 114: 1–36.

Kulak, Osman, Yusuf Sahin, and Mustafa Egemen Taner. 2012. “Joint Order Batching and Picker Routing in Single and Multiple-cross-aisle Warehouses Using Cluster-based Tabu Search Algorithms.” *Flexible Services and Manufacturing Journal* 24: 52–80.

Letchford, Adam N., Saeideh D. Nasiri, and Dirk Oliver Theis. 2013. “Compact Formulations of the Steiner Traveling Salesman Problem and Related Problems.” *European Journal of Operational Research* 228 (1): 83–92.

Li, Jianbin, Rihuan Huang, and James B. Dai. 2017. “Joint Optimisation of Order Batching and Picker Routing in the Online Retailers Warehouse in China.” *International Journal of Production Research* 55 (2): 447–461.

Manzini, Riccardo. 2012. *Warehousing in the Global Supply Chain*. London: Springer.

Marchet, Gino, Marco Melacini, and Sara Perotti. 2015. “Investigating Order Picking System Adoption: A Case-study-based Approach.” *International Journal of Logistics Research and Applications* 18 (1): 82–98.

Masae, Makusee, Christoph H. Glock, and Eric H. Grosse. 2020. “Order Picker Routing in Warehouses: A Systematic Literature Review.” *International Journal of Production Economics* 224: 107564.

Padberg, Manfred, and Giovanni Rinaldi. 1991. “A Branch-and-Cut Algorithm for the Resolution of Large-Scale Symmetric Traveling Salesman Problems.” *SIAM Review* 33 (1): 60–100.

Petersen, Charles. 1997. “An Evaluation of Order Picking Routing Policies.” *International Journal of Operations & Production Management* 17: 1098–1111.

Ratliff, H. Donald, and Arnon S. Rosenthal. 1983. “Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem.” *Operations Research* 31 (3): 507–521.

Roodbergen, Kees Jan, and René de Koster. 2001. “Routing Methods for Warehouses with Multiple Cross Aisles.” *International Journal of Production Research* 39 (9): 1865–1883.

Scholz, André, Sebastian Henn, Meike Stuhlmann, and Gerhard Wässcher. 2016. “A New Mathematical Programming Formulation for the Single-Picker Routing Problem.” *European Journal of Operational Research* 253 (1): 68–84.

Scholz, A., and G. Wässcher. 2017. “Order Batching and Picker Routing in Manual Order Picking Systems: the Benefits of Integrated Routing.” *Central European Journal of Operations Research* 25: 491–520.

Theys, Christophe, Olli Bräysy, Wout Dullaert, and Birger Raa. 2010. “Using a TSP Heuristic for Routing Order Pickers in Warehouses.” *European Journal of Operational Research* 200 (3): 755–763.

Tompkins, J., J. White, Yavuz Bozer, and J. Tanchoco. 2010. *Facilities Planning*, New Jersey: John Wiley & Sons.

Tsai, C.-Y., J. J. H. Liou, and T.-M. Huang. 2008. “Using a Multiple-GA Method to Solve the Batch Picking Problem: Considering Travel Distance and Order Due Time.” *International Journal of Production Research* 46 (22): 6533–6555.

Valle, Cristiano, John Beasley, and Alexandre Cunha. 2016. “Modelling and Solving the Joint Order Batching and Picker Routing Problem in Inventories.” In *Combinatorial Optimization. ISCO 2016. Lecture Notes in Computer Science*. Vol. 9849, edited by R. Cerulli, S. Fujishige, and A. Mahjoub. Cham: Springer.

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.

van Gils, Teun, An Caris, Katrien Ramaekers, and Kris Braeckers. 2019. “Formulating and Solving the Integrated Batching, Routing, and Picker Scheduling Problem in a Real-life Spare Parts Warehouse.” *European Journal of Operational Research* 277 (3): 814–830.

van Gils, Teun, Katrien Ramaekers, An Caris, and René B. M. de Koster. 2018. “Designing Efficient Order Picking Systems by Combining Planning Problems: State-of-the-art Classification and Review.” *European Journal of Operational Research* 267 (1): 1–15.

Vaughan, T. S. 1999. “The Effect of Warehouse Cross Aisles on Order Picking Efficiency.” *International Journal of Production Research* 37 (4): 881–897.

Wässcher, Gerhard. 2004. “Order Picking: A Survey of Planning Problems and Methods.” In *Supply Chain Management and Reverse Logistics*, edited by H. Dyckhoff, R. Lacks, and J. Reese. Berlin: Springer.

Won, J., and S. Olafsson. 2005. “Joint Order Batching and Order Picking in Warehouse Operations.” *International Journal of Production Research* 43 (7): 1427–1442.
## Appendix

### Notation Explanation

| Sets | Explanation |
|------|-------------|
| \(T\) | set of available trolleys |
| \(O\) | set of orders |
| \(L_o\) | set of picking locations of order \(o\) |
| \(V\) | set of all locations |
| \(V_L\) | set of picking locations |
| \(V_I\) | set of artificial locations |
| \(V_{sub}(i)\) | set of picking locations within subaisle \(i\) |
| \(\tilde{E}\) | set of directed edges connecting neighboring locations |
| \(\tilde{E}'\) | set of directed edges connecting neighboring artificial locations while ignoring picking locations |
| \(\delta^+(S)\) | set of directed edges with one end in set \(S\) |
| \(\delta^-(S)\) | set of directed edges in \(\tilde{E}\) that leave/enter set \(S\) |

| Constants | Explanation |
|-----------|-------------|
| \(W_{sub}\) | the number of subaisles |
| \(s\) | the origin of the warehouse |
| \([u, v]\) | the ordered pair of location \(u\) and location \(v\), which represents a directed edge |
| \((u, v)\) | the unordered pair of location \(u\) and location \(v\), which represents an undirected edge |
| \(f(i) / (i)\) | the northern/southern artificial location of subaisle \(i\) |
| \(n(v) / s(v)\) | the adjacent northern/southern location of \(v\) |
| \(Q_{N}(v) / Q_{S}(v) / Q_{E}(v) / Q_{W}(v)\) | the adjacent northern/southern/eastern/western artificial location of artificial location \(v\) |
| \(b_o\) | size of order \(o\) |
| \(B\) | available capacity of a trolley |

| Variables | Explanation |
|-----------|-------------|
| \(x_{tuv}\) | Binary variable that takes value 1 if and only if \((u, v) \in \tilde{E}\) is traversed by walk \(t\) |
| \(y_{tv}\) | Binary variable that takes value 1 if and only if trolley \(t\) visits location \(v\) |
| \(z_{ot}\) | Binary variable that takes value 1 if and only if trolley \(t\) picks order \(o\) |
| \(\alpha_{tv} / \beta_{tv}\) | Binary variable that takes value 1 only if there exists a straight path connecting the northern/southern artificial location and \(v\) in walk \(t\) |
| \(\gamma_{tuv}\) | Continuous variable that indicate the volume of flow from artificial location \(v_0\) passing through arc \((u, v) \in \tilde{E}\) in walk \(t\) |

| Formulation | Constraints | Explanation |
|-------------|-------------|-------------|
| For Analysis | \(P_{sub}\) (19)–(25) | feasible region of subaisle cuts |
| | \(P_{basic}\) (9)–(18) | the basic formulation for the JOBPRP |
| | \(P_{A}\) (9)–(25) | the basic formulation with subaisle cuts |
| | \(P_{G}\) (26)–(42) | a formulation which only force artificial locations to be in the same connected component |
| | \(P_{F}\) (26)–(34), (36)–(46) | a flow-based formulation which only force artificial locations to be in the same connected component |
| | \(P_{G}\) (19)–(42) | a non-compact improved formulation for the JOBPRP |
| | \(P_{F}\) (19)–(34), (36)–(46) | a flow-based improved formulation for the JOBPRP |
| | \(P_{U}\) (52)–(63) | a TSP-based no-reversal formulation for a single-block warehouse |
| | \(P_{G}\) (64)–(74) | a TSP-based no-reversal formulation for a 2-block warehouse |
| For Experiment | \(P_{O}\) | \(P_{basic}\) with the valid inequalities defined in Valle, Beasley, and Salles da Cunha \(2017\) and column inequalities |
| | \(P_{G}\) | \(P_{G}\) with aisles cuts, artificial vertex reversal constraints and column inequalities |
| | \(P_{F}\) | \(P_{F}\) with artificial vertex reversal constraints and column inequalities |
| | \(P_{U}\) | the no-reversal formulation in Valle, Beasley, and Salles da Cunha \(2017\) with column inequalities |
| | \(P_{G}\) | \(P_{G}\) with column inequalities |
| | \(P_{F}\) | \(P_{F}\) with column inequalities |
Table A1. Detailed results for the basic cuts.

| Δ | O | \( P_{O} + \text{basic cuts} \) | \( P_{G}^{+} + \text{basic cuts} \) | \( P_{I}^{+} + \text{basic cuts} \) |
|---|---|---|---|---|
| 5 | 5 | 0.34 | 346 | 346 | 0 | 0.21 | 346 | 346 | 0 | 0.4 | 346 | 346 | 0 |
| 10 | 6.96 | 578 | 578 | 0 | 3.23 | 578 | 578 | 0 | 66 | 578 | 578 | 0 |
| 15 | 26 | 650 | 650 | 0 | 9.4 | 650 | 650 | 0 | 46 | 650 | 650 | 0 |
| 16 | 406 | 766 | 766 | 0 | 37 | 766 | 766 | 0 | 29 | 766 | 766 | 0 |
| 17 | 239 | 802 | 802 | 0 | 46 | 802 | 802 | 0 | 281 | 802 | 802 | 0 |
| 18 | 2400 | 870 | 806 | 7.4 | 95 | 840 | 840 | 0 | 830 | 840 | 840 | 0 |
| 19 | 2020 | 856 | 856 | 0 | 72 | 856 | 856 | 0 | 605 | 856 | 856 | 0 |
| 20 | 2067 | 864 | 864 | 0 | 75 | 864 | 864 | 0 | 745 | 864 | 864 | 0 |
| 21 | 2400 | 902 | 846 | 6.2 | 270 | 892 | 892 | 0 | 2400 | 892 | 892 | 0 |
| 22 | 2400 | 892 | 886 | 1.1 | 190 | 892 | 892 | 0 | 1045 | 892 | 892 | 0 |
| 23 | 2400 | 918 | 868 | 5.5 | 416 | 908 | 908 | 0 | 1465 | 908 | 908 | 0 |
| 24 | 2400 | – | 733 | – | 270 | 892 | 892 | 0 | 2400 | 892 | 850 | 4.7 |
| 25 | 2400 | 1112 | 890 | 20 | 2400 | 1108 | 944 | 14.8 | 2400 | 1120 | 839 | 25.1 |
| 30 | 2400 | 1232 | 844 | 31.5 | 2400 | 1212 | 942 | 22.3 | 2400 | 1202 | 862 | 28.3 |
| 10 | 5 | 0.38 | 368 | 368 | 0 | 0.12 | 368 | 368 | 0 | 0.09 | 368 | 368 | 0 |
| 15 | 122 | 874 | 874 | 0 | 36 | 874 | 874 | 0 | 224 | 874 | 874 | 0 |
| 16 | 138 | 926 | 926 | 0 | 65 | 926 | 926 | 0 | 430 | 926 | 926 | 0 |
| 17 | 408 | 960 | 960 | 0 | 97 | 960 | 960 | 0 | 501 | 960 | 960 | 0 |
| 18 | 410 | 970 | 970 | 0 | 151 | 970 | 970 | 0 | 1055 | 970 | 970 | 0 |
| 19 | 300 | 978 | 978 | 0 | 156 | 978 | 978 | 0 | 1421 | 978 | 978 | 0 |
| 20 | 524 | 984 | 984 | 0 | 166 | 984 | 984 | 0 | 1640 | 984 | 984 | 0 |
| 21 | 1410 | 990 | 990 | 0 | 281 | 990 | 990 | 0 | 2192 | 990 | 990 | 0 |
| 22 | 2203 | 1000 | 1000 | 0 | 125 | 1000 | 1000 | 0 | 2400 | 1014 | 938 | 7.5 |
| 23 | 2400 | 1344 | 743 | 10.7 | 2400 | 1132 | 1011 | 10.7 | 2400 | 1152 | 926 | 19.6 |
| 24 | 2406 | 1184 | 947 | 44.4 | 2400 | 1166 | 1056 | 9.4 | 2402 | 1194 | 937 | 21.5 |
| 25 | 2400 | 1230 | 945 | 23.2 | 2400 | 1220 | 991 | 18.8 | 2400 | 1196 | 962 | 19.6 |
| 30 | 2400 | 1306 | 958 | 26.6 | 2400 | 1286 | 1052 | 18.2 | 2400 | 1288 | 954 | 25.9 |
| 20 | 5 | 13 | 570 | 570 | 0 | 1.29 | 570 | 570 | 0 | 8.7 | 570 | 570 | 0 |
| 15 | 57 | 912 | 912 | 0 | 47 | 912 | 912 | 0 | 113 | 912 | 912 | 0 |
| 16 | 2400 | 1022 | 1022 | 1 | 63 | 1022 | 1022 | 0 | 385 | 1022 | 1022 | 0 |
| 17 | 2400 | 1200 | 1088 | 9.3 | 377 | 1200 | 1200 | 0 | 1607 | 1200 | 1200 | 0 |
| 18 | 2400 | 1282 | 1047 | 18.3 | 1602 | 1250 | 1250 | 0 | 2400 | 1250 | 1183 | 5.4 |
| 19 | 2400 | 1330 | 1093 | 17.8 | 2400 | 1296 | 1216 | 6.2 | 2400 | 1302 | 1160 | 10.9 |
| 20 | 2400 | 1342 | 1188 | 11.5 | 1799 | 1304 | 1304 | 0 | 2400 | 1322 | 1187 | 10.2 |
| 21 | 2400 | 1620 | 903 | 44.3 | 2400 | 1520 | 1193 | 21.5 | 2400 | 1518 | 1036 | 31.8 |
| 22 | 2400 | 1820 | 966 | 46.9 | 2400 | 1532 | 1180 | 23 | 2400 | 1536 | 1107 | 27.9 |
| 23 | 2400 | – | 989 | – | 2400 | 1598 | 1175 | 26.5 | 2400 | 1618 | 1044 | 35.5 |
| 24 | 2400 | – | 916 | – | 2400 | 1640 | 1185 | 27.7 | 2400 | 1652 | 1097 | 33.6 |
| 25 | 2400 | 1736 | 987 | 33.1 | 2400 | 1674 | 1164 | 30.5 | 2400 | 1654 | 1086 | 34.3 |
| 30 | 2400 | – | 979 | – | 2400 | 1934 | 1104 | 42.9 | 2400 | 1940 | 970 | 50 |
| Δ   | O | T(seconds) | UB | LB | GAP(%) | T(seconds) | UB | LB | GAP(%) | T(seconds) | UB | LB | GAP(%) |
|-----|---|------------|----|----|--------|------------|----|----|--------|------------|----|----|--------|
| 5   | 5 | 0.17       | 346 | 346 | 0      | 0.17       | 346 | 346 | 0      | 0.38       | 346 | 346 | 0      |
| 10  | 5.3 | 578 | 578 | 0 | 2.16 | 578 | 578 | 0 | 27 | 578 | 578 | 0 |
| 15  | 18 | 650 | 650 | 0 | 17 | 650 | 650 | 0 | 36 | 650 | 650 | 0 |
| 16  | 22 | 766 | 766 | 0 | 33 | 766 | 766 | 0 | 128 | 766 | 766 | 0 |
| 17  | 1460 | 802 | 802 | 0 | 40 | 802 | 802 | 0 | 208 | 802 | 802 | 0 |
| 18  | 1681 | 840 | 840 | 0 | 68 | 840 | 840 | 0 | 781 | 840 | 840 | 0 |
| 19  | 2400 | 872 | 831 | 4.7 | 78 | 856 | 856 | 0 | 363 | 856 | 856 | 0 |
| 20  | 2400 | 888 | 755 | 15 | 139 | 864 | 864 | 0 | 701 | 864 | 864 | 0 |
| 21  | 2400 | 898 | 871 | 3 | 161 | 892 | 892 | 0 | 1880 | 892 | 892 | 0 |
| 22  | 2400 | 898 | 857 | 4.6 | 180 | 892 | 892 | 0 | 735 | 892 | 892 | 0 |
| 23  | 2400 | 908 | 908 | 0 | 251 | 908 | 908 | 0 | 1385 | 908 | 908 | 0 |
| 24  | 2400 | 1112 | 933 | 12.1 | 128 | 802 | 802 | 0 | 363 | 856 | 856 | 0 |
| 25  | 2400 | 1136 | 840 | 840 | 0 | 161 | 892 | 892 | 0 | 1880 | 892 | 892 | 0 |
| 26  | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 |
| 27  | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 |
| 28  | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 |
| 29  | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 |
| 30  | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 | 2400 | 2400 | 2400 | 0 |

| Δ | O | T(seconds) | UB | LB | GAP(%) | T(seconds) | UB | LB | GAP(%) | T(seconds) | UB | LB | GAP(%) |
|---|---|------------|----|----|--------|------------|----|----|--------|------------|----|----|--------|
| 5 | 5 | 8.9 | 570 | 570 | 0 | 570 | 570 | 0 | 4.4 | 570 | 570 | 0 |
| 10 | 125 | 912 | 912 | 0 | 21 | 912 | 912 | 0 | 157 | 912 | 912 | 0 |
| 15 | 2400 | 1022 | 1008 | 1.4 | 68 | 1022 | 1022 | 0 | 315 | 1022 | 1022 | 0 |
| 16 | 2400 | 1238 | 1045 | 15.6 | 530 | 1200 | 1200 | 0 | 2394 | 1200 | 1200 | 0 |
| 17 | 2400 | 1282 | 938 | 26.8 | 1266 | 1250 | 1250 | 0 | 2400 | 1250 | 1190 | 4.8 |
| 18 | 2400 | 1318 | 1187 | 9.9 | 1568 | 1288 | 1288 | 0 | 2400 | 1308 | 1076 | 17.7 |
| 19 | 2400 | 1350 | 1114 | 17.5 | 2400 | 1316 | 1251 | 4.9 | 2400 | 1308 | 1168 | 10.7 |
| 20 | 2400 | 1360 | 1072 | 21.2 | 2400 | 1332 | 1264 | 5.1 | 2400 | 1358 | 1109 | 18.3 |
| 21 | 2400 | 1744 | 1014 | 41.9 | 2400 | 1510 | 1187 | 21.4 | 2400 | 1542 | 1094 | 29.1 |
| 22 | 2400 | 1590 | 999 | 37.2 | 2400 | 1534 | 1206 | 21.4 | 2400 | 1566 | 1116 | 28.7 |
| 23 | 2400 | 1742 | 1020 | 41.4 | 2400 | 1606 | 1165 | 27.5 | 2400 | 1618 | 1117 | 31 |
| 24 | 2400 | 1672 | 940 | 43.8 | 2400 | 1660 | 1143 | 31.1 | 2400 | 1664 | 1133 | 31.9 |
| 25 | 2400 | – | 917 | – | 2400 | 1680 | 1191 | 29.1 | 2400 | 1672 | 1060 | 36.6 |
| 30 | 2400 | – | 967 | – | 2400 | 1896 | 1134 | 40.2 | 2400 | 1944 | 987 | 49.2 |

**Table A2.** Detailed results for the single traversing constraints.