The joint order batching and picker routing problem : modelled and solved as a clustered vehicle routing problem

The joint order batching and picker routing problem: modelled and solved as a clustered vehicle routing problem

Babiche Aerts[1] * Trijntje Cornelissens [1] Kenneth Sörensen [1]

[1]ANT/OR - Operations Research Group, Department of Engineering Management, University of Antwerp, Prinsstraat 13, 2000 Antwerp, Belgium

Abstract

The joint order batching and picker routing problem (JOBPRP) is a promising approach to minimize the order picking travel distance in a picker-to-parts warehouse environment. In this paper, we show that the JOBPRP can be modelled as a clustered vehicle routing problem (CluVRP), a variant of the capacitated VRP in which customers are grouped into clusters. To solve this cluster-based model of the JOBPRP, we apply a two-level variable neighborhood search (2level-VNS) metaheuristic, previously developed for the CluVRP, and study which adaptations are required for it to perform efficiently in a warehouse environment. Additionally, we evaluate if the Hausdorff distance used as an approximation for the clusters' proximity in the CluVRP, performs equally well when determining closeness between pick orders in a warehouse. We compare the performance of the Hausdorff-based batching criterion to the cumulative minimal aisles visited-criterion, known as a well-performing batching metric in rectangular warehouses with parallel aisles.

The 2level-VNS performs well compared to state-of-the-art algorithms specifically developed for the order batching problem (OBP) in a single-block warehouse. A multi-start VNS remains slightly superior to our approach. Concerning the Hausdorff distance, we conclude that in most experiments, the minimum-aisles criterion retains a better fit in the warehouse context.

Keywords: Order batching, Picker routing, Vehicle routing, Variable neighborhood search

1 Introduction

Order picking, the act of retrieving Stock Keeping Units (SKUs) from storage locations to fulfil order requests, is the most costly operation in warehouse management

*corresponding author: babiche.aerts@uantwerpen.be

(Petersen and Schmenner, 1999). This is especially true for picker-to-parts systems where the pickers walk through aisles, search and pick items, and bring them to a depot for consolidation. With the advent of e-commerce, a shift has occurred from unit-load (pallet) orders to customer orders consisting of various SKUs in small quantities (Van Gils et al., 2018), creating even more movement in the warehouse. Certainly in those e-commerce environments, walking is by far the most time consuming activity (De Koster et al., 2007) and the minimisation of pickers' travel distance is an objective pursued in both academia and in practice.

The pickers' travel distance is affected by tactical decisions such as the layout of the warehouse (*warehouse design*), storage location of the items (*storage assignment*) and assignment of pickers to specific picking areas (*zoning*). The act of clustering customer orders into batches (*batching*) and sequencing the picking of items (*routing*), on the other hand, are considered main factors to influence the picking performance at operational level (Yu and De Koster, 2009). Since B2C e-commerce orders are generally small, pickers' travel distances can be reduced by combining, i.e., batching, multiple orders in one pick tour instead of picking them separately. This gives rise to two optimization problems, both pursuing a minimisation of the total travel distance: how to combine orders into batches, and how to sequence the pick operations for each batch. Since large savings on the walking distance can be realised by solving the batching and routing problem simultaneously (Van Gils et al., 2018), we focus in this paper on the combined problem, known as the joint order batching and picker routing problem (JOBPRP).

Just as the *picker routing problem* (PRP) bears large similarities with the travelling salesman problem (TSP), we observe that the JOBPRP closely resembles the clustered VRP (CluVRP), a variant of the capacitated VRP. The capacitated VRP aims to assign the delivery packages of customers to vehicles of a specific capacity such that the total travel distance of the vehicles is minimised. In the CluVRP, introduced by Sevaux et al. (2008), customers are partitioned into clusters based on a predefined criterion (e.g., postal code), with the additional constraint that customers belonging to the same cluster need to be visited by the same vehicle. By replacing vehicles by batches, clusters by orders, and customers by pick operations, the JOBPRP can be modelled as the CluVRP. This structural overlap between vehicle routing and order picking was highlighted early in the literature by Chisman (1975) who introduces the idea of a clustered TSP (CTSP). Since then, further studies on the cluster-based models are rather found within the VRP literature, apart from Löffler et al. (2018) who exploit the similarities between the CTSP and the PRP to plan the routing in an AGV-assisted order picking system.

In this paper, we model the JOBPRP as the CluVRP, and study the two-level variable neighborhood search (2level-VNS) metaheuristic, previously developed for the CluVRP by Defryn and Sörensen (2017), as a solution method for the JOBPRP. This two-level approach alternates between a VNS at the order level to assign orders to batches, and a VNS at pick operation-level to construct the routes. In the original algorithm, Defryn and Sörensen (2017) utilize the Hausdorff distance at the cluster level (respectively, order level) to determine how far two customer clusters (respectively, orders) are located from each other, and consequently decide which clusters are combined in one trip. The sequence of customers (respectively, pick locations) in a trip

is not optimized until the second level of the VNS, and differs from VNS algorithms developed by Albareda-Sambola et al. (2009) and Menéndez et al. (2017). These algorithms optimize the batch composition and routes simultaneously and require the assumption of a routing heuristic to do so, which not always guarantees the shortest tour. In this paper, we test if the Hausdorff distance, and by extension the two level approach, proposed for the CluVRP performs equally well for the JOBPRP by comparing it to state-of-the-art algorithms specifically developed for the *order batching problem* (OBP).
The remainder of this paper is organised as follows. In section 2 we describe in detail the JOBPRP and highlight the similarities and differences with the CluVRP. Section 3 presents a literature review on the OBP, PRP, JOBPRP and CluVRP. In section 4, we introduce the Hausdorff distance as used for the CluVRP, and suggest improvements to the implementation in case of the JOBPRP. In section 5 we describe the 2level-VNS algorithm in detail. The experimental setup and computational results, as well as a comparison with state-of-the-art OBP algorithms, are discussed in section 6. We conclude and elaborate on future research in section 7.

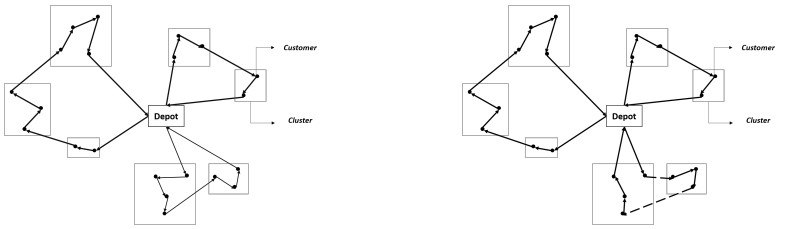
2 The joint order batching and picker routing problem

2.1 Problem description

In an e-commerce environment, a customer order typically consists of one or multiple order lines that each contain a particular item and a requested quantity. These items are stored at different pick locations, according to a predefined storage policy. Pickers are provided with a pick list with all order lines that must be handled and the particular sequence of pick locations (i.e., picking route). To avoid additional sorting operations and associated costs at the depot, items that belong to the same order are not split over different pick lists but forced to be processed together (*order integrity rule*). Consequently, a *batch* is defined as the set of complete orders processed in the same pick tour. The capacity of a batch is defined by the number of items that fit into the batch (to the example of Gibson and Sharp (1992), Zhang et al. (2017) and Scholz and Wäscher (2017)). The pick tours always start and end at the (single) depot.
Given the objective of minimizing the total travel distance, the batch capacity and all orders that have to be picked, two sub-questions remain:

- How are orders combined into batches?
- For each batch, in which sequence does the picker visit the pick locations that store the items requested by the orders in the batch?

When both questions are treated in an integrated way, this gives rise to the combined problem, known as the JOBPRP.



(a) Clustered VRP with hard cluster constraints **(b)** Clustered VRP with soft cluster constraints

Figure 1: Representation of the clustered VRP with (a) hard and (b) soft cluster constraints. Soft cluster constraints allow to visit clusters multiple times if this leads to shorter routes, shown by the dashed line at the bottom.

2.2 Comparison with the clustered VRP

In the CluVRP, customers are clustered according to a specific criterion such as geography (e.g., same postal code) or preference (e.g., preferring the same driver). These clusters are assigned to vehicles while respecting the vehicle's capacity, that is, a cluster can only be assigned to a vehicle if the demand of all customers in the cluster fits in the vehicle. Once all clusters are assigned, a route is constructed for each vehicle such that all customers are served and the total travel distance is minimized (Defryn and Sörensen, 2017).

Barthélemy et al. (2010) introduce a heuristic approach to solve the CluVRP where all customers of the same cluster are forced to be served before the vehicle moves on to the next cluster. This problem is referred to as the *clustered VRP with strong cluster constraints*. However, for distance purposes it could be better to relax this rule and allow vehicles to enter and leave clusters multiple times. Customers of the same cluster, however, are still to be visited by the same vehicle. Defryn and Sörensen (2017) introduced this variant as *the clustered VRP with soft cluster constraints*. Both variants of the CluVRP are illustrated in fig. 1a and fig. 1b.

The CluVRP with soft cluster constraints shares its mathematical structure with the JOBPRP. Indeed, the JOBRPR can be modelled on an undirected graph $G = (V, E)$, where V represents the set of pick operations to be executed. A pick operation V_i is characterized by an item requested by a specific order. The quantity of the item to be retrieved by the pick operation V_i is denoted by q_i. Items from the same storage location but requested by different orders are included as separate pick operations, and as such as separate nodes. Node V_0 refers to the depot, visited at the start and end of each pick route. For each edge $(i, j) \in E$ that connects two nodes (i.e., pick operations), the distance d_{ij} is defined as the shortest travel distance between the locations related to pick operation V_i and pick operation V_j. For pick operations V_i and V_j that request the same item stored at the same location but for different orders, the distance d_{ij} is equal to 0.

K is a set of homogeneous batches, each with a capacity Q, defined as the total number of items that can be picked by a batch. The set of orders is given by R. The set of pick operations to complete an order r is presented by $C_r = \{V_i \in V \backslash V_0 : r_i = r\}$. S is any subset of V that is not equal to V, $\delta^+(S)$ is the set of outgoing edges

$(i,j) \in S \times V\backslash S$ and $\delta^-(S)$ the set of incoming edges $(i,j) \in V\backslash S \times S$. The mathematical model is described as follows:

$$x_{ijk} = \begin{cases} 1, & \text{if the route for batch } k \text{ goes from the location to perform pick operation } i \text{ to the} \\ & \text{location to perform pick operation } j \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ik} = \begin{cases} 1, & \text{if pick operation } i \text{ is performed by batch } k \\ 0, & \text{otherwise} \end{cases}$$

The objective function of the JOBPRP is modelled as

$$Min \sum_{(i,j)\in E} \sum_{k\in K} d_{ij} x_{ijk} \tag{1}$$

Subject to

$$\sum_{k\in K} y_{ik} = 1 \qquad \forall i \in V\backslash V_0 \tag{2}$$

$$\sum_{k\in K} y_{0k} = \sum_{j\in V\backslash V_0} \sum_{k\in K} x_{0jk} \leq |K| \tag{3}$$

$$\sum_{j\in V} x_{ijk} = \sum_{j\in V} x_{jik} = y_{ik} \qquad \forall k \in K, \forall i \in V \tag{4}$$

$$\sum_{i\in V\backslash V_0} q_i y_{ik} \leq Q \qquad \forall k \in K \tag{5}$$

$$\sum_{i\in S} \sum_{j\notin S} x_{ijk} \geq y_{hk} \qquad \forall S \subseteq V\backslash V_0, \forall h \in S, \forall k \in K \tag{6}$$

$$\sum_{(i,j)\in\delta^+(C_r)} \sum_{k\in K} x_{ijk} = \sum_{(i,j)\in\delta^-(C_r)} \sum_{k\in K} x_{ijk} \geq 1 \qquad \forall r \in R \tag{7}$$

$$y_{ik} = y_{jk} \qquad \forall i,j \in C_r, \forall r \in R, \forall k \in K \tag{8}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in V, \forall j \in V, \forall k \in K \tag{9}$$

$$y_{ik} \in \{0,1\} \qquad \forall i \in V, \forall k \in K \tag{10}$$

The objective function (1) minimizes the total travel distance over all batches. Constraints (2) guarantee that each pick operation is executed by one batch only. Constraint (3) forces all batches, that perform at least one pick operation, to start their tour at the depot. Constraints (4) ensure each location related to a specific pick operation is accessed and left by the same batch. Constraints (5) state that the capacity of the batch cannot be exceeded by the number of items related to the pick operations performed by that batch. The subtour elimination constraints are described by (6). Constraints (7) refer to the soft cluster constraints. Lastly, the order integrity rule is represented by constraints (8).

CluVRP with soft cluster constraints		JOBPRP
Concepts		
Vehicle	$\leftrightarrow$	Batch
Cluster	$\leftrightarrow$	Order
Customer	$\leftrightarrow$	Pick operation
Input data & precomputation		
List of customer clusters, for each cluster: - Cluster demand = sum of customers demand - Customers' coordinates	$=$	List of orders, for each order: - Order quantity = sum of requested item quantities - Items' number
Routing environment - Available road network	$\leftrightarrow$	Warehouse layout - Warehouse layout structured by aisles - Item storage locations
Cluster - Assignment to vehicle / to batch		
Clusters are combined and assigned to a vehicle	$=$	Orders are combined and assigned to a batch
Customers of same cluster visited by same vehicle	$=$	Order integrity rule
Each customer visited only once	$\leftrightarrow$	Items can reoccur in multiple orders
Cluster demand cannot exceed vehicle capacity	$=$	Order quantity cannot exceed batch capacity
Route construction for vehicle / for batch		
Create route for each vehicle that minimizes the total travel distance	$=$	Create route for each batch that minimizes the total travel distance
Start and end route at depot	$=$	Start and end route at depot

Table 1: Comparison of the CluVRP with soft cluster constraints and the JOBPRP.

Apart from terminology-based adaptations, the mathematical structure of the JOBPRP and the CluVRP with soft cluster constraints is identical. Differences between both problems are rather reflected in the input data. A first dissimilarity is the warehouse layout, which requires aisles to enter and exit to move from one pick location to the next. To our knowledge, a similar routing environment has not yet been considered in the CluVRP. This dissimilarity, however, does not require any adaptations to the mathematical model. Instead, we find all information regarding the warehouse layout and its aisle-structure to be reflected in the distance matrix. A second difference, is that the CluVRP not allows to visit a customer more than once, while in the JOBPRP multiple orders can request the same item, stored at the same location. In the mathematical formulation no constraint has to be adapted or added to deal with this disparity. Instead, we define V as the set of pick operations rather than pick locations, where two pick operations can refer to the retrieval of the same item at the same location but for different orders. A summary of similarities and differences between the CluVRP and JOBPRP is presented in table 1.

3 Literature review

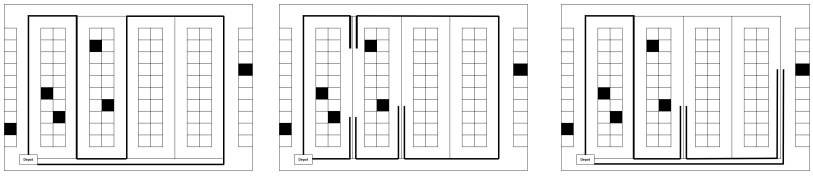
In section 3.1, we give a literature overview for the OBP and PRP. In the following sections, we summarise the literature on the JOBPRP (section 3.2) and CluVRP (section 3.3).

3.1 The order batching and picker routing problem

The order batching problem (OBP) and the picker routing problem (PRP) have been extensively studied in the warehouse literature as two separate problems. Because

both problems are strongly connected (Van Gils et al., 2018), assumptions have to be made in order to solve the OBP or PRP as individual problems: the PRP cannot be solved without knowing the outcome of the OBP, i.e., which items to pick. The OBP, on its own turn, cannot be solved without the assumption of a routing strategy. It is clear that larger distance savings could be realised if the batching and routing decisions were optimized simultaneously. However, research on the joint problem is not found in the warehouse literature until 2005, discussed in section 3.2, and the majority of studies have focused on either the PRP or OBP. We give an overview of both problems and proposed solution methods.

Knowing which items to collect and the pick locations where to find them, the PRP aims to determine the sequence in which pick locations are visited, while minimizing the total travel distance of the pick tour (Scholz and Wäscher, 2017). The PRP in a rectangular warehouse with parallel aisles, proven to be NP-hard (Won and Olafsson, 2005), is often solved using heuristics dedicated to the typical aisle-structure. Such heuristics give straightforward guidelines (e.g. "traverse the aisle if a pick location has to be visited") which are considered easy to memorize and execute, although they not always result in the best, i.e., shortest, route. Among them, the *s-shape* (or *traversal*) (fig. 2a), *largest gap* (fig. 2b) and *combined* routing heuristic (fig. 2c) are implemented most frequently. For a full description of these dedicated routing heuristics we refer to De Koster et al. (2007).



(a) S-shape routing **(b)** Largest gap routing **(c)** Combined routing

Figure 2: Visulation of routing heuristics dedicated to warehouse layout, illustrated for a warehouse layout with parallel aisles.

Ratliff and Rosenthal (1983) developed an exact algorithm to solve the PRP in a single-block warehouse with parallel aisles, and is later extended to a two-block warehouse by Roodbergen and Koster (2001). For larger multi-block layouts, Cambazard and Catusse (2018) solve the PRP in a warehouse with up to eight cross-aisles using a dynamic programming approach, although the complexity is exponential in the number of cross-aisles. An alternative solution method for the PRP is available as the PRP shares many similarities with the well-studied travelling salesman problem (TSP). Indeed, the Lin-Kernighan-Helsgaun (LKH) algorithm, considered one of the best heuristics to solve the TSP, outperforms the warehouse dedicated routing heuristics and has proven to solve the PRP close to optimality (Theys et al., 2010); (Van Gils et al., 2018). Despite their good performance, we observe the use of these more complex routing heuristics in the warehouse literature to be limited. Gademann and Velde (2005) argue that these algorithms result in illogical routes and inconvenient for pickers. Scholz and Wäscher (2017) deem this argument to be valid for a single-block warehouse but not when more complex warehouse layouts are considered in which

also simple, dedicated routing heuristics become less straightforward.
Knowing the maximum capacity of the batch, the list of orders, and the routing strategy to be used, the OBP aims to combine orders into batches that require a minimal total travel distance. Gademann and Velde (2005) have proven that the OBP is NP-hard and polynomially solvable when batches consist of only two orders. Consequently, a large share of OBP research is devoted to the study of heuristics to solve the problem for realistic instances. The few proposed exact solution approaches are limited to small problem instances (up to 32 orders in the case of Gademann and Velde (2005)). Öncan (2015) introduced Mixed Integer Linear Programming (MILP) formulations in three variants, where each variant considers another routing strategy (s-shape, return and midpoint routing).
Batching heuristics to solve the OBP can be distinguished into five groups: priority rule-based algorithms (e.g., First-Come-First-Serve (FCFS) rule), seed algorithms, savings algorithms, metaheuristics, and data mining approaches. We refer the reader to the detailed reviews of De Koster et al. (2007) and Henn and Wäscher (2012) and supplement with further references to contributions we deem relevant for this paper.
Seed algorithms create batches one by one, in which the choice of the next order to be assigned is based on a chosen batching criterion. Often different criteria are used to select the *seed*, i.e., first order of the batch, and the additional, or *accompanying*, orders (De Koster et al., 2007). It is common to assign orders to a particular batch until its remaining capacity is insufficient to include the smallest unassigned order of the instance. Ho et al. (2008) present an extensive study in which 11 seed-order criteria and 14 accompanying-order criteria were tested. Many of these batching criteria utilize a metric to approximate the closeness between orders instead of considering the actual distance between pick locations. Moreover, we observe many of these criteria make use of the characteristics related to the warehouse layout. For instance, the number of aisles integrated in the minimum aisles visited batching criterion, which combines orders into batches such that the total number of aisles visited is minimized.
Seed algorithms can be implemented in two ways. In *single mode*, the seed and accompanying orders in a batch are treated as individual orders. In *cumulative mode*, the seed order is renewed every time an order is added to the batch (De Koster et al., 1999). We illustrate for the aisle-based batching criterion. Imagine two orders, each requesting items from the first two aisles. The total number of aisles visited in single mode is four, ignoring the overlap of aisles, while in cumulative mode, the total is equal to two. The frequent renewals of the seed order make the cumulative mode more complex and time consuming, but in general, the cumulative information positively influences the outcome. Ho and Tseng (2006), Ho et al. (2008) and Van Gils et al. (2018) study the cumulative minimal aisles visited-criterion, which proved to work well in comparison to other studied batching criteria. De Koster et al. (1999), however, showed that the cumulative mode outperformed the single mode only to a minor extent for other studied batching criteria, e.g., the maximum aisles visited-criterion.
Savings algorithms are based on the Clarke and Wright algorithm, originally developed to solve VRPs. The algorithm initially assigns each order to a separate batch. In a second stage, orders are merged if a distance saving can be realised. This move is evaluated using a savings matrix, which is often only calculated once, referred to as the C&W(i) variant of the savings algorithm (Van Gils et al., 2018). Instead of travel

distance, the savings concept can be applied to other metrics, illustrated by Rosenwein (1996). The authors consider the minimum aisles visited-criterion, for which the merge of orders is beneficial if less aisles have to be visited.
Metaheuristics to solve the OBP have been proposed in the warehouse literature since 2005. Many of these heuristics create an initial solution using the savings algorithm discussed above (Henn and Wäscher, 2012). Next, the metaheuristic aims to improve the solution by neighborhood exploration through which alternative batch assignments are found and evaluated. Different types of metaheuristics have been proposed: genetic algorithms (Hsu et al., 2005), variable neighborhood search algorithms (Albareda-Sambola et al., 2009); (Menéndez et al., 2017), tabu search algorithms (Öncan, 2015) and attribute-based hill climber algorithms (Henn and Wäscher, 2012). Among them, the multi-start VNS approach by Menéndez et al. (2017), is considered the state of the art solution method for the OBP.

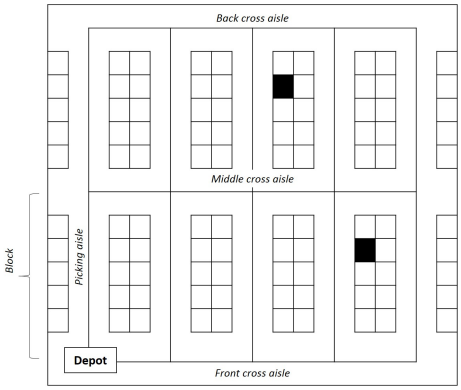


Figure 3: Illustration of a parallel, two-block warehouse.

Most OBP metaheuristics have in common to explore the neighborhood of the incumbent solution through the execution of moves which are generally accepted when the total travel distance is improved. This total travel distance is computed assuming dedicated routing heuristics, described earlier. Some OBP metaheuristics use more advanced routing heuristics (e.g., an alternative routing algorithm based on the combined routing strategy proposed by Menéndez et al. (2017)) than others, but all have in common that the choice of routing is fixed which might obstruct to solve the OBP to optimality. Moreover, Roodbergen (2001) showed that the performance of dedicated routing algorithms tends to deteriorate when altering the number of cross aisles. Indeed, the majority of studies on the OBP perform experiments for a parallel single-block warehouse (fig. 2), rather than multiple-block layouts (fig. 3) or warehouses with an atypical layout. For such warehouses, the performance of these OBP metaheuristics is unknown.

3.2 The joint order batching and picker routing problem

Because of the strong connection between the OBP and PRP, studies have started to focus on the joint problem, known as the JOBPRP (Briant et al., 2020). In the JOBPRP, the batching and routing decisions are optimized in a simultaneous way, and differs from the OBP where the routing strategy is considered to be known in advance and fixed throughout the algorithm. Methods to solve the JOBPRP mainly include meta-heuristics, summarized below.

Won and Olafsson (2005) are one of the first to study the JOBPRP, and propose a heuristic based on the savings concept while the corresponding PRPs are solved with a two-opt heuristic. Kulak et al. (2012) use a tabu search algorithm based on a similarity-regret value index (RS-RV) that defines the overlap in travel distance if orders are merged. The resulting PRPs are solved by two TSP heuristics. Tsai et al. (2008) present a multiple genetic algorithm approach, one to compose the batches, one to construct the routes. Cheng et al. (2015) propose a hybrid approach in which a particle swarm optimization is used for the order allocation, while an ant colony optimization algorithm determines the route for each batch. Scholz and Wäscher (2017) present an iterated local search approach in which they propose a routing heuristic derived from the exact solution approach presented by Roodbergen and Koster (2001). Briant et al. (2020) propose a heuristic based on column generation, providing lower and upper bounds for JOBPRP instances that take place in a rectangular warehouse. The authors state that the PRP on itself can be easily solved to optimality by specifically making use of the properties of a rectangular warehouse layout. So far, Valle et al. (2017) are the only ones to develop an exact approach for the JOBPRP. With a branch-and-cut algorithm, they are able to solve instances up to 20 orders to optimality.

3.3 The clustered vehicle routing problem

The CluVRP is an extension of the clustered TSP (CTSP), introduced by Chisman (1975). In the CTSP, a route is determined that visits all customers which have been assigned to predetermined clusters. Both the customer sequence within a cluster and the sequence of clusters are optimized to minimize the length of the route. The author shows that the cluster concept can be extended to the warehousing context, and presents a MILP formulation to solve the clustered PRP for a one batch-problem. The MILP forces orders to be handled one by one, currently known as the strong cluster constraint variant of the problem (visualised in fig. 1a for the CluVRP). Recently, Löffler et al. (2018) revived the application of the CTSP with strong cluster constraints in the field of warehousing. The authors study a picking environment in which pickers are assisted by an AGV (automated guided vehicle). The AGV is utilized as pick cart and autonomously follows the picker during his pick tour. Once the order is completed, the AGV returns to the depot while the picker resumes his pick tour assisted by a new, empty AGV.

Despite the early application of the CTSP in the warehousing context, we find further research on the CTSP, the CluVRP, and in general, cluster-based routing problems, mostly to be dedicated to the field of vehicle routing. Among these problems, we find the Generalised Vehicle Routing Problem (GVRP), introduced by Ghiani and Im-

	Exact algorithm	Heuristic algorithm
Clustered Traveling Salesman Problem (CTSP)		
Introduced by Chisman (1975)		Chisman (1975) (warehouse application)
		Laporte et al. (1997)
		Ding et al. (2007)
		Löffler et al. (2018) (warehouse application)
Generalised Vehicle Routing Problem (GVRP)		
Introduced by Ghiani and Improta (2000)	Pop et al. (2012)	Pop et al. (2011)
	Bektaş et al. (2011)	Pop et al. (2013)
Clustered Vehicle Routing Problem (CluVRP)		
Introduced by Sevaux et al. (2008)		
Hard-cluster constraint variant	Battarra et al. (2014)	Barthélemy et al. (2010)
		Pop and Chira (2014)
		Vidal et al. (2015)
		Marc et al. (2015)
		Hintsch and Irnich (2018)
		Decomposition-based approach:
		Defryn and Sörensen (2015)
		Expósito-Izquierdo et al. (2016)
		Defryn and Sörensen (2017)
		Horvat-Marc et al. (2018)
		Pop et al. (2018)
Soft-cluster constraint variant		
Introduced by Defryn and Sörensen (2017)	Hintsch and Irnich (2020)	**Defryn and Sörensen (2017)**
		Hintsch et al. (2019)
Selective Vehicle Routing Problem (SVRP)		
Introduced by Posada et al. (2018)	Posada et al. (2018)	Posada et al. (2019)
	Sabo et al. (2020)	

Table 2: Literature overview of the clustered vehicle routing problem, and related problems.

prota (2000). In the GVRP, customers are assigned to a predetermined cluster which is visited exactly once. The demand of the entire cluster is delivered at the location of just one customer, included in the respective cluster. Which customer this is, will be determined while optimizing the inter-cluster routes over all vehicles. The CluVRP, proposed by Sevaux et al. (2008), is an extension of the CTSP to multiple vehicles, and a variant of the GVRP since each customer has to be visited. Both inter- and intra-cluster routes have to be optimized. The latest addition to the pool of cluster-based routing problems, is the Selective VRP (SVRP), presented by Posada et al. (2018). The SVRP is a variant of the GVRP in which customers are allowed to be assigned to more than one cluster. Through the optimization of the inter-cluster routes, and while respecting the vehicles' capacity, one determines which customers to visit, two customers each belonging to another cluster or one customer shared by both clusters, and in which order. In table 2 we present a non-exhaustive overview of research on the cluster-based routing problems discussed above, categorised by the proposed solution method (exact versus heuristic algorithm). Due to the NP-hardness of each discussed cluster-based routing problem, we find the majority of papers to focus on a heuristic solution approach.

In the current paper, we study whether the two-level VNS developed by Defryn and Sörensen (2017) (highlighted in bold in table 2) can be used to solve the JOBPRP. This paper advances from previous research by Defryn and Sörensen (2015), and is con-

sidered the first to utilize a two-level (or decomposition-based) approach to solve the strong cluster constraint CluVRP. On the first level, the authors focus on the assignment of clusters to vehicles as well as the cluster sequence within each vehicle. A VNS is applied to find a local optimum. On this level, the travel distance between customers is not considered. Instead, the authors propose an approximation for the closeness between clusters to make the evaluation of vehicle assignments and cluster sequences easier. The optimization of the customer sequence is thereby postponed to a second level. On this second level, the route for each vehicle is optimized while considering the travel distance between customers. Again, a VNS is utilized to find a local optimum, while respecting the strong cluster constraints. The algorithm continues to iterate between both levels until a stopping criterion is met. To determine the closeness between clusters, i.e., the criterion applied in the VNS at the cluster level, the authors propose the clusters' Euclidean center.

The idea to decompose the CluVRP into two subproblems is further explored by Expósito-Izquierdo et al. (2016). They propose a combination of the record-to-record travel algorithm using the clusters' Euclidean centres to solve the subproblem at cluster-level and the LKH algorithm to solve the routing problem within the clusters. Horvat-Marc et al. (2018) and Pop et al. (2018), on the other hand, both present a genetic algorithm to solve the subproblem at cluster-level. At customer-level, Horvat-Marc et al. (2018) propose a simulated annealing approach, while Pop et al. (2018) utilize the TSP concorde solver (an online solver tool, available at `http://www.math.uwaterloo.ca/tsp/concorde.html`) to optimize the intra-cluster routes.

Both Defryn and Sörensen (2015) and Expósito-Izquierdo et al. (2016) propose the clusters' Euclidean center as an approximation for the closeness between customer clusters, which restricts their approaches to VRPs defined by Euclidean distances. As an alternative, Defryn and Sörensen (2017) suggest to use the Hausdorff distance as closeness criterion. The Hausdorff distance is often used for object matching in the field op computer vision and object recognition (Sim et al., 1999) to measure the similarities between two sets (Hung and Yang, 2004). The Hausdorff distance can be applied to problems defined by Euclidean (e.g., VRP), as well as Manhattan distances (e.g., PRP).

The solution methods discussed above tackle the CluVRP with strong cluster constraints. Defryn and Sörensen (2017) acknowledge that, if the situation allows, travel distances can be reduced by no longer obliging clusters to be visited consecutively. They introduce the problem as the *soft cluster constraints CluVRP* and solve it with their two-level VNS approach developed for the strong-constraint variant, with only minor adaptations. For all instance classes, the authors are able to reduce the total travel distance when relaxing the cluster constraints. Hintsch and Irnich (2020) confirmed that this relaxation enables a reduced total travel distance and quantified this reduction at 6.21% for medium-sized instances. A heuristic approach to solve the soft cluster constraints CluVRP is presented by Hintsch et al. (2019). The authors adopt the large multiple neighborhood search algorithm developed for the strong cluster constraints CluVRP ((Hintsch and Irnich, 2018)), and acknowledge that major modifications are in order to fit the algorithm to the soft-constraint characteristics of the problem.

Even though other heuristic approaches exist for both the strong- and soft-cluster

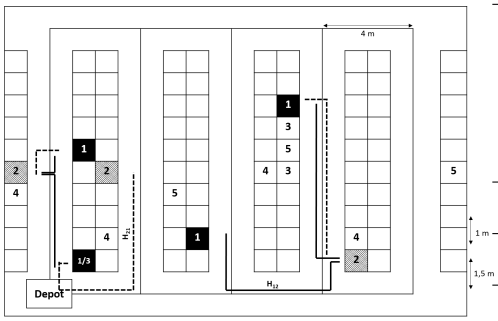
constraints CluVRP, we chose to solve the JOBPRP by means of the two-level VNS approach proposed by Defryn and Sörensen (2017). We deem the use of the Hausdorff distance as approximation for inter-cluster distances, an interesting and unique feature of this approach. The two-level VNS offers a good setting to experiment with this measure of approximation in the warehousing context as other batching criteria can be implemented as benchmark without interfering with the further course of the algorithm.

4 Proposed order batching heuristics

In this section we describe three different constructive heuristics to assign orders to batches. One will be chosen later to integrate in the 2level-VNS to solve the JOBPRP. First, we describe the batching heuristic based on the Hausdorff distance as originally developed by Defryn and Sörensen (2017) (section 4.1). In section 4.2, we propose adaptations to the original batching heuristic to align better with the characteristics of the JOBPRP while keeping the Hausdorff distance as batching criterion. To evaluate both batching heuristics and the Hausdorff distance as batching criterion, we include the aisle-based batching heuristic as benchmark. The aisle-based heuristic is described in detail in section 4.3.

4.1 Original Hausdorff-based batching heuristic - HausOrig

The *one-way Hausdorff distance* is calculated for each pair of orders (r_i, r_j) as follows: for each pick location to be visited for order r_i, select the closest (i.e., for which the shortest distance is travelled) pick location visited for order r_j. The largest of these distances is the one-way Hausdorff distance between r_i and r_j (Hung and Yang, 2004). We illustrate the calculation of the one-way Hausdorff distance on a five-order example, visualised in fig. 4a. Each cell refers to a pick location, the number in the cell indicates the order(s) requesting the item stored at the respective pick location. We compute the one-way Hausdorff distance between r_1 and r_2 by determining for each pick location visited for r_1, the closest pick location visited for r_2, visualised by the thick full lines. Of the four distances, the longest (8 m) is the one-way Hausdorff distance between r_1 and r_2, indicated by H_{12}.
The one-way Hausdorff distance is not necessarily symmetrical. Consider the one-way Hausdorff distance between r_2 and r_1. The dashed lines in fig. 4a show for each pick location visited for r_2 the closest pick location visited for r_1. The travel distance marked H_{21} is the longest (11 m) and differs from the 8 m distance found for H_{12}. In the field of object recognition it is common to take the maximum of both values to define the Hausdorff distance between two sets, such that the direction no longer matters (Hung and Yang, 2004). Similarly, we apply this definition when computing the Hausdorff distance between two orders in future experiments. Consequently, the Hausdorff distance between r_1 and r_2 is 11 m. The Hausdorff distance between each pair of orders is saved in the symmetrical interorder distance matrix (fig. 4b).
Defryn and Sörensen (2017) describe the following Hausdorff-based constructive heuristic, from now on referred to as the *HausOrig* batching heuristic. Prior to the assign-



(a) JOBPRP example

	r_0	r_1	r_2	r_3	r_4	r_5
r_0	/	20.5	13.5	19.5	14.5	21.5
r_1	20.5	/	11	12	8	15
r_2	13.5	11	/	11	11	14
r_3	19.5	12	11	/	15	15
r_4	14.5	8	11	15	/	14
r_5	21.5	15	14	15	14	/

Nb requested items

r_0	r_1	r_2	r_3	r_4	r_5
/	4	3	3	4	3

(b) Interorder distance matrix and order size

Figure 4: JOBPRP example (a) illustrating the computation of the Hausdorff distance between order 1 and order 2. The numbers in the cells represent the items requested by the orders. Corresponding interorder distance matrix, based on the Hausdorff distance, and orders' sizes are presented on the right.

ment, orders are sorted in decreasing order according to size (number of requested items (see fig. 4b)), without further distinction between equally sized orders. For the example in section 4.1, orders are sorted as follows: r_4, r_1, r_5, r_3, r_2. At this stage, also the number of available batches is known, which is the minimal number necessary to have all orders assigned to a batch (discussed in more detail in section 5.1). The capacity of the batch is given and equal for all batches. For the five order-example, we compute two available batches, each with a capacity of 12 items. Both batches are considered when deciding to which batch the next order is assigned. The first order on the list, r_4, is assigned to the batch with sufficient remaining capacity, and for which the last added order is closest to r_4, i.e., smallest Hausdorff distance. For empty batches the depot is considered as the last added order. Next, r_1 is added to Batch 1 as r_1 is considered to be closer to r_4 than to the depot. The procedure is repeated until all orders are assigned, resulting in the order assignment presented in table 3.

	Order assignment	**Hausdorff distance batch**
Batch 1	0 4 1 5 0	14.5 + 8 + 15 + 21.5 = **59 m**
Batch 2	0 3 2 0	19.5 + 11 + 13.5 = **44 m**
Total original Hausdorff distance		**103 m**
Batch capacity = 12 items		

Table 3: Order assignment for the example in fig. 4a, determined by the HausOrig batching heuristic.

Because the HausOrig heuristic assigns orders primarily on a size-based criterion, the potential of the Hausdorff distance as a batching criterion might not be fully exploited. Moreover, the HausOrig heuristic only considers the last order added to each batch to select the batch for the next order. In the JOBPRP, the sequence of orders in the assignment should no longer matter, which creates opportunities currently not embraced. We propose a variant of the HausOrig heuristic in the following section.

4.2 Adapted Hausdorff-based batching heuristic - HausAdap

Given the remarks in the previous section, we propose an alternative constructive heuristic which we deem to align better with the characteristics of the JOBPRP. We do not adapt the Hausdorff-based batching criterion, but rather make adaptations to the heuristic such that it is used to its full potential. We refer to this batching heuristic as *HausAdap*, of which the outline is shown in Algorithm 1.

Algorithm 1 HausAdap batching heuristic

1: **Input:** Set of unassigned orders R with $r \in \{1, ..., n\}$, batch capacity Q
2: **Step 0: Precomputation**
3: $H[0...n][0...n] \leftarrow$ interorder (Hausdorff) distance matrix;
4: $K \leftarrow$ minimal number of batches;
5: **for each** $k \leq K$ **do**
6: Initialize batch k (assign depot, remaining capacity $\leftarrow Q$, total Hausdorff distance $\leftarrow 0$);
7: **end for**
8: **Step 1: Assignment**
9: $k \leftarrow 1$;
10: **while** R is non empty **do**
11: **if** batch k only includes depot **then**
12: Assign $r \in R$ to batch k for which $H[0][r]$ is the smallest*;
13: Update remaining capacity, increase total Hausdorff distance batch k by $H[0][r]$;
14: Remove r from R;
15: **else if** remaining capacity batch $k \geq$ size of the smallest order $\in R$ **then**
16: *bestHausDist* $\leftarrow \infty$, $r_{best} \leftarrow \emptyset$;
17: **for each** $s \in$ batch $k \backslash$depot **do**
18: Find $r \in R$ for which $H[s][r]$ is the smallest* and for which size $\leq$ remaining capacity batch k;
19: *bestHausDist* $\leftarrow min\{$*bestHausDist*;$H[s][r]\}$, update r_{best} if necessary;
20: **end for**
21: Assign r_{best} to batch k;
22: Update remaining capacity, increase total Hausdorff distance batch k by *bestHausDist*;
23: Remove r_{best} from R;
24: **else if** remaining capacity batch $k <$ size of the smallest order $\in R$ **then**
25: Find $r \in$ batch $k \backslash$depot for which $H[0][r]$ is the largest;
26: Increase total Hausdorff distance batch k by $H[0][r]$;
27: $k \leftarrow k + 1$;
28: **end if**
29: **end while**
30: *In case of a tie, choose r that requests the most number of items

A first modification we propose, is to fill batches one by one instead of considering all batches at once. In the HausAdap heuristic, orders are added to the current batch as long as the smallest unassigned order fits into the remaining capacity (line 15 in Algorithm 1). Secondly, we omit the sorting operation and use the Hausdorff distance as primary criterion to assign orders to batches, rather than order size. In case of a tie, preference goes to the largest order, with the argument that with the limited capacity

it is easier to assign larger orders first.
The third adjustment is to extend the pool of information when deciding which order will be assigned next. Since the JOBPRP resembles the soft cluster constraints variant of the CluVRP, one can benefit from this property by considering the Hausdorff distance of all orders already assigned to the batch (lines 17-20 in Algorithm 1), instead of considering only the last one. The depot, however, is not considered, unless the seed-order is selected (lines 11-14 Algorithm 1). We argument that a visit to the depot is obliged anyway, and it is more relevant to take into account only fellow orders when selecting the next order to add to the batch. With a similar reasoning we determine the Hausdorff distance to return to the depot, that is when the batch is full or has insufficient remaining capacity. However, instead of the smallest we consider the largest of all Hausdorff distances between the depot and any order included in the batch as experiments showed a clear preference for the latter option (lines 25-28 in Algorithm 1).
We apply the HausAdap batching heuristic to the example of fig. 4a, which results in the following order assignment:

	Order assignment	**Hausdorff distance batch**
Batch 1	0 2 1 4 0	13.5 + 11 + 8 + 20.5 = **53 m**
Batch 2	0 3 5 0	19.5 + 15 + 21.5 = **56 m**
Total adapted Hausdorff distance		**109 m**
Batch capacity = 12 items		

Table 4: Order assignment for the example in fig. 4a, determined by the HausAdap batching heuristic.

Both the HausOrig and HausAdap heuristic are implemented in single rather than cumulative mode, explained in section 3.1. In the cumulative mode it is unclear how to correctly handle the final Hausdorff distance of a batch given that its orders are treated as one, cumulative order. The Hausdorff distance of the batch would become the travel distance between the farthest location visited by the batch and the depot, ignoring all intermediate visited locations and geographical similarities between orders. This would make the Hausdorff distance deviate from its original definition. Together with former mentioned time-based arguments (discussed in section 3.1), we decide to implement both Hausdorff-based batching heuristics in single mode.

4.3 Aisle-based batching heuristic - Aisles

With the HausOrig and HausAdap batching heuristics, we are the first, to our knowledge, to propose the minimal Hausdorff distance as a batching criterion in the warehouse literature. To evaluate this new batching criterion, we include the minimal aisles visited-criterion as benchmark, proven in prior OBP studies to be a performant batching criterion. We briefly explain the aisle-based batching heuristic, referred to as *Aisles*.
The Aisles heuristic is implemented as seed-algorithm (batch per batch) in cumulative mode. The order requiring the least number of aisles is selected as seed. Accompanying orders are chosen based on the least number of *additional* aisles to visit on top

of the ones already accessed for the batch. Only the number of aisles is taken into account; information regarding the distance between aisles is not considered.

	assigned \ to add	r_0	r_1	r_2	r_3	r_4	r_5
Stage 1	r_0	/	3	3	2	4	3
Stage 2	r_3	/	1	1	/	2	2
Stage 3	r_3, r_1	/	/	1	/	1	2

Table 5: The interorder distances in terms of the additional aisles to visit, worked out for the example in fig. 4a.

With the calculations provided in table 5, we illustrate the Aisles heuristic for the example given in fig. 4a. In contrast to the interorder distance matrix showed in fig. 4b, we find the structure of the matrix to be different when the Aisles heuristic is applied. Due to the cumulative implementation of the Aisles heuristic, the values of the matrix are no longer fixed, but change according to the orders already assigned to the batch. In the first stage, with no order other than the depot assigned yet, the values indicate for each order the number of aisles requiring a visit. Given that the order with the smallest number is selected, r_3, we compute for each remaining order the additional number of aisles to visit in the second stage. The final batch assignment is presented in table 6.

	Order assignment	**Number of aisles**
Batch 1	0 3 1 4 0	4 aisles
Batch 2	0 2 5 0	4 aisles
Total number of aisles		**8 aisles**
Batch capacity = 12 items		

Table 6: Order assignment for the example illustrated in fig. 4a, determined by the Aisles batching heuristic.

5 Metaheuristic approach for the joint order batching and picker routing problem

In this paper, we solve the JOBPRP with the two-level variable neighborhood search (2level-VNS) algorithm proposed by Defryn and Sörensen (2017). At order-level, orders are assigned to batches using one of the batching criteria presented in section 4. Next, batches of orders are passed on to the pick operation-level, where a route is constructed for each batch. At both levels, a variable neigborhood search (VNS) algorithm is used to find a local optimum, although at both levels another objective, minimal value with respect to the chosen batching criterion versus minimal travel distance, is pursued. A full description of the algorithm is given in the following sections.

5.1 Step 1: Precomputation

For each instance, the following information is known:

- Warehouse layout (e.g., parallel aisles), number and width of aisles, number and width of cross-aisles, width and depth of pick locations, length of the rack.
- List of pick locations of the items: each pick location is dedicated to one SKU and each SKU is located at a single pick location (no scattered storage).
- Capacity of the batch: equal for all batches, defined by number of items.
- List of orders, with for each order the list of items and required quantity.

During precomputation, the shortest travel distance between pick locations (including the depot) is calculated. Interorder distances are computed according to the chosen batching criterion, illustrated in fig. 4b and table 5.
Given the capacity of the batch and the total number of items requested, the minimum number of batches is defined. De Koster et al. (1999) show that the number of batches used and the total travel distance are strongly related, confirmed by Menéndez et al. (2017) who obtained a shorter total travel distance when less batches are used. We determine the number of batches K as follows:

$$K = \left\lceil \frac{\text{Total number of items}}{\text{Capacity}} \right\rceil$$

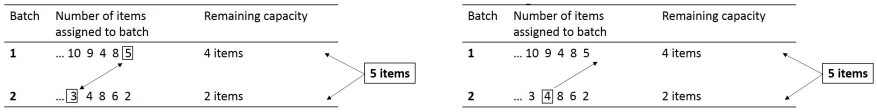
K is currently a lower bound on the number of batches and does not guarantee that the order integrity rule is obeyed. We describe in section 5.3 how K is updated if necessary.

5.2 Step 2: Construction phase

An initial assignment of orders to batches is generated according to the HausOrig (section 4.1), HausAdap (section 4.2) or Aisles batching heuristic (section 4.3). A feasible solution on order level is found if all orders are assigned to a batch.

5.3 Step 3: Redistribution phase

During the construction phase, it occurs that not all orders fit into the predefined number of batches K. In that case, the redistribution function is called to free capacity by swapping two orders of different batches (i.e., *inter batch swap*, visualised in fig. 5a) or to reallocate an order to a another batch (i.e., *inter batch relocation*, visualised in fig. 5b). Both operators are detailed in table 7, with n referring to the number of orders. During redistribution, orders are reshuffled to gain capacity. The orders' size is the primary criterion to do so, while the batching criterion not matters in this stage. The move that leads to sufficient capacity release or to the largest capacity release (in case multiple redistributions are required to create enough space in a batch) is executed. We remove the respective order(s) from its current batch and add it last in the sequence



(a) Example of an inter batch swap

(b) Example of an inter batch relocation

Figure 5: Illustrations of an inter batch swap (a) and relocation (b) during redistribution to free capacity for an order of five items.

Inter batch operators	
Swap	Swap two orders belonging to two different batches. Each order is added last in the sequence of the other batch. Complexity: O(n^2)
Relocation	Remove an order from one batch, add it last in the sequence of another batch. Complexity: O(n)

Table 7: Operators performed during redistribution at the order level. The same operators are used during intensification at order-level when the HausAdap or Aisles batching heuristic is implemented.

of the other batch. To avoid moves being reversed directly after execution, a simple tabu list is included which prevents the last move to be undone.
After 10 consecutive reassignments with insufficient capacity saving, the number of batches K is increased by one. All orders are reverted back to the status 'unassigned' and we restart the construction from scratch.

5.4 Step 4: Intensification at the order level

The initial batch assignment is based on a greedy heuristic that makes the best local choice at each step rather than looking at the entire solution. During the intensification phase, we aim to improve the batch assignment by a VNS algorithm at the order level, based on the batching criterion chosen at the start.
In the original CluVRP approach with the HausOrig heuristic implemented, the sequence of orders in a batch is of great importance. Therefore, it is relevant to not only explore inter batch moves (e.g., swap orders between batches), but also to consider intra batch moves that seek to optimize the order sequence within a batch. Defryn and Sörensen (2017) propose five local search operators that explore inter and intra batch neighbourhoods, presented in table 8. In table 9 we show an example of such a move starting from the initial assignment determined by the HausOrig heuristic, provided in table 3. Order r_1 is relocated from Batch 1 to the first position in the sequence of Batch 2. The difference in total Hausdorff distance, represented by delta, is positive and gives a negative advice regarding this move.

Intra batch operators	
Swap	Swap the position of two orders in the sequence of the same batch.
Relocate	Remove an order from a sequence, insert it at another position in the sequence of the same batch.
Two-Opt	Remove two edges from the batch's sequence, replace them by two new edges.
Inter batch operators	
Swap	Swap two orders belonging to two different batches.
Relocation	Remove an order from one batch, insert it at a position in the sequence of another batch.

Table 8: Moves at the order level during intensification, when implementing the HausOrig batching heuristic. All operators have complexity O(n^2).

	Order assignment	**Hausdorff distance batch**
Batch 1	0 4 5 0	14.5 + 14 + 21.5 = **50 m**
Batch 2	0 1 3 2 0	20.5 + 12 + 11 + 13.5 = **57 m**
Delta Batch 1	- 8 - 15 + 14	
Delta Batch 2	- 19.5 + 20.5 + 12	
Total original Hausdorff distance	Delta = 4 m	**107 m**
Batch capacity = 12 items		Do not perform move

Table 9: Illustration of an inter batch relocation when the HausOrig heuristic is implemented.

In section 4.2, we proposed an alternative Hausdorff-based constructive heuristic. Suggestions were introduced to benefit more from the JOBPRP characteristics, and continue to apply during intensification. Therefore, we present the following adaptations to the intensification at order-level when the HausAdap heuristic is implemented. First, we no longer perform intra batch operators. We argue that the optimization of the orders' sequence has no influence on the routes of the batches composed afterwards. Aligned with the former argument, we decide to no longer swap or relocate orders to a specific position in an order sequence, but move them to the end of the order sequence. The operators performed during intensification at order-level are listed in table 7, which are the same operators used during redistribution. These operators are also applicable when the Aisles heuristic is implemented, for which similar arguments hold because of its cumulative implementation.
Despite the fact that less moves have to be evaluated, we find that the corresponding time benefit compensates only partly for the complexity of the HausAdap implementation. This is because in the HausAdap heuristic all orders are considered when determining the Hausdorff distance of the batch. Consequently, the removal of one order can have a large impact on the batch's Hausdorff distance, illustrated in table 10. We show an example of an inter batch relocation performed on the initial order assignment determined with the HausAdap heuristic (see table 4). Order r_1 is removed from Batch 1 and added last in the sequence of Batch 2, which reduces the number of relocation possibilities (before r_3, before r_5, last in the sequence) to only one. In contrast to the HausOrig implementation, more computational effort is required to compute the Hausdorff distance of the modified batches. For instance, to return to the depot, the largest Hausdorff distance between the depot and any order included in the batch is considered. For Batch 1, that was 21.5 m, the Hausdorff distance be-

tween the depot and r_1. With the removal of r_1, we have to find again the largest Hausdorff distance between the depot and any of the remaining orders in Batch 1.

	Order assignment	Hausdorff distance batch
Batch 1	0 2 4 0	13.5 + 11 + 14.5 = **39 m**
Batch 2	0 3 5 1 0	19.5 + 15 + 12 + 21.5 = **68 m**
Delta Batch 1	- 8 - 20.5 + 11 + 11 + 14.5	
Delta Batch 2	12	
Total adapted Hausdorff distance	Delta = -2 m	**107 m**
Batch capacity = 12 items		Perform move

Table 10: Illustration of an inter batch relocation when the HausAdap heuristic is implemented.

During intensification at the order level, a move is accepted if the value of the chosen batching criterion improves. The neighborhoods are checked in a random way by the algorithm and for each neighborhood all possible moves are evaluated. If an improvement is found, the algorithm returns to the first neighborhood. Otherwise, the algorithm randomly picks one of the remaining neighborhoods. The intensification at the order level terminates when all neighborhoods were consecutively checked with no improvement.

5.5 Step 5: Conversion from order to pick operation-level

For each batch constructed in the previous stage, a routing is determined in which all pick locations required to fulfil the orders of a batch, are included. This routing problem is solved as a TSP by means of a greedy heuristic: the pick location with the shortest travel distance to the previous pick location is visited next. The procedure is repeated until all pick locations to be visited are included.

5.6 Step 6: Intensification at the pick operation level

The intensification process at the pick operation level aims to improve the routes composed in the previous stage. Again, a VNS is applied and follows the same procedure as described in section 5.4. Both intra and inter batch operators, described in table 11, are implemented, which are no longer evaluated by the chosen batching criterion but by total travel distance.
Apart from intra batch operators, also inter batch moves at pick operation-level are included by Defryn and Sörensen (2017). This means that orders are swapped or relocated between batches if it results in a reduced total travel distance. In some way it seems strange to include these moves since much attention already has been paid to swapping and relocating orders at the order level. However, these moves have only been evaluated using the chosen batching criterion, while further travel distance improvements could be found when the position of and distance between pick locations is taking into account.
At this point, the algorithm has produced an *initial solution*. The total travel distance of the solution, used later to evaluate the performance of the algorithm, cumulates the travel distance over all batches.

Intra batch operators	
Swap	Swap the position of two pick operations in a single pick tour.
Relocate	Remove a pick operation, insert it at another position in the same pick tour.
Two-opt	Remove the edges between two pick operations, replace them by two new edges.
Inter batch operators	
Swap	Remove two orders' pick operations from the sequence, insert them at the best position in the sequence of the other batch.
Relocation	Remove an order's pick operation from the sequence, insert it at the best position in the sequence of another batch.

Table 11: Description of operators implemented during intensification at the pick operation level. All operators have complexity O(n^2).

5.7 Step 7: Diversification phase and iterative loop

After the initial solution has been obtained, the algorithm continues to explore the solution space through diversification. Part of the solution is destroyed and subsequently repaired to comply with feasibility rules. Because of the order integrity rule, we perform the perturbation at order-level to ensure orders remain complete. From each batch, a number of orders is removed (we currently set this number to 10% of the orders included in a batch) and reassigned to batches randomly, while meeting the capacity constraint. Redistribution (described in section 5.3) is performed when no sufficient remaining capacity is available to fit all orders.
Steps 4 to 6 of the algorithm are repeated for the newly composed batches. In case a better solution is found, the incumbent solution is updated. We currently apply the same stopping criterion as Defryn and Sörensen (2017): the algorithm terminates after 1000 consecutive iterations without improvement.

6 Numerical experiments

In this section we present the experimental results to evaluate the performance of the proposed 2level-VNS algorithm. All experiments are conducted on the instance set developed by Henn and Wäscher (2012). The characteristics of the dataset are described in section 6.1. In section 6.2, we report preliminary tests on the configuration of our 2level-VNS algorithm, with in particular adaptations required to obtain good results in an acceptable time.
Finally, we analyse the outcome of the 2level-VNS approach in a twofold way:

- We compare the results for the HausOrig, HausAdap and Aisles heuristic to conclude which batching heuristic is superior, and whether this depends on features such as the number of orders or the batch size (section 6.3).
- We compare the performance of the 2level-VNS with algorithms considered state-of-the-art in the OBP literature (section 6.4).

We have implemented the 2level-VNS in C++ Visual Studio 17. All experiments are carried out on an Intel(R) Core i7-6820HQ CPU, 2.7 GHz laptop with 16 GB RAM of memory. A peak performance of 132,06 GFlops was reached by our laptop in the

LINPACK benchmark (8 threads, LinX program version 0.6.5). For all experiments the average of three runs is reported.

6.1 Instance description

The dataset of Henn and Wäscher (2012) originally includes 5760 instances, of which half follow an ABC storage policy (ABC) and half a random distribution (Ran). As we did not explicitly include decisions related to the storage allocation, we currently focus only on instances following a random distribution. Of these 2880 instances, Menéndez et al. (2017) report the results for a limited set, which they found to be a representative subset. This set includes 32 instances, referred to as *Ran_32*.
In the Ran_32 dataset, half of the instances are originally labelled as 's-shape instances', half as 'largest gap instances'. When comparing both subsets in terms of various parameters (number of orders, number of total items requested, number of unique items requested), we were not able to explain in what way both instance groups differ, apart from their solution method which we do not value important during instance generation. In further experiments, we make no distinction and report the accumulated results.
All instances are defined on a warehouse with a single-block layout, consisting of 10 aisles. Each aisle contains 90 pick locations, 45 on each side, leading to 900 pick locations in total. Each pick location has a width of 1 m. It takes 5 m to move from one aisle to the next, and 1.5 m to move from the depot to the first pick location.
Instances differ in terms of the number of orders (n) and the batch capacity (C). For each (n,C)-combination, of which the values can be found in table 12, two instances are included in the Ran_32 dataset. For all instances, the number of items per order are uniformly distributed in U{5,25}.

Number of orders (n)	**Batch capacity (C)**
40	30
60	45
80	60
100	75

Table 12: Overview of (n,C)-values represented in instances of Ran_32 dataset.

Although the 2level-VNS is currently tested for a single-block warehouse, our approach can easily be extended to instances defined on a warehouse with a multi-block layout, as well as layouts other than the often used parallel warehouse layout.

6.2 Preliminary experiments to speed up the algorithm

We test the 2level-VNS algorithm on the 32 instances of Ran_32 by alternately implementing the HausOrig, HausAdap and Aisles batching heuristic. Regardless of the batching heuristic, we find the computation time to be extremely high in comparison to state-of-the-art references. On average, the 2level-VNS requires 11 times more computation time than the multi-start VNS by Menéndez et al. (2017). Since the JOBPRP

is an operational problem, the instances should be solved in a realistic time, which is currently not guaranteed. Adaptations to the algorithm are necessary to make it competitive to available (J)OBP(RP) algorithms.
We propose three adaptations:

- **Omit the inter batch moves at pick operation-level**: during the VNS at pick operation-level, only intra batch moves are evaluated and performed. Swapping and relocating orders between batches will be reserved for the VNS performed at the order level.
- **Omit the diversification stage**: we take the initial solution as the final solution.
- **Adapt the stopping criterion of the algorithm**: the algorithm terminates after 60 seconds. This replaces the original stopping criterion of 1000 consecutive iterations without improvement, counted once an initial solution was found.

All adaptations are tested separately on the Ran_32 instances and evaluated in table 13. We report for each adaptation the average percentage deviation relative to the original algorithm implementation (all stages included and with the original stopping criterion), together with the time improvement realised by the adaptation. Each suggestion, independent of the implemented batching heuristic, reduces the computation time significantly. However, with a switch to a time-based stopping criterion we find the solution quality to deteriorate only little ($< 0.48\%$). We therefore decide to continue experiments with the 2level-VNS in which a time-based stopping criterion is adopted.

	Avg. gap total travel distance (%)			**Avg. gap computation time (%)**		
	HausOrig	HausAdap	Aisles	HausOrig	HausAdap	Aisles
1) Algorithm without inter batch moves at routing-level						
Ran_32	3.63%	3.13%	3.10%	-94.42%	-93.81%	-93.95%
2) Algorithm without diversification stage						
Ran_32	3.28%	2.70%	2.70%	-99.92%	-99.92%	-99.92%
3) Algorithm with time-based stopping criterion (60 seconds)						
Ran_32	0.48%	0.42%	0.44%	-77.48%	-78.97%	-76.64%

Table 13: Evaluation of three alternatives to speed up the 2level-VNS. Values represent the average gap (%) in respect of the solution and computation time obtained by the 2level-VNS algorithm with the original stopping criterion.

To conclude about the duration adopted in this time-based stopping criterion, we perform additional experiments. Figure 6 shows for a random chosen instance how the solution's quality evolves in time when solved with the 2level-VNS with original stopping criterion in which respectively the HausOrig, HausAdap or Aisles heuristic is implemented. Each mark refers to a point in time when an improved total travel distance is found. On the y-axis one reads the solution's improvement (in %) relative to

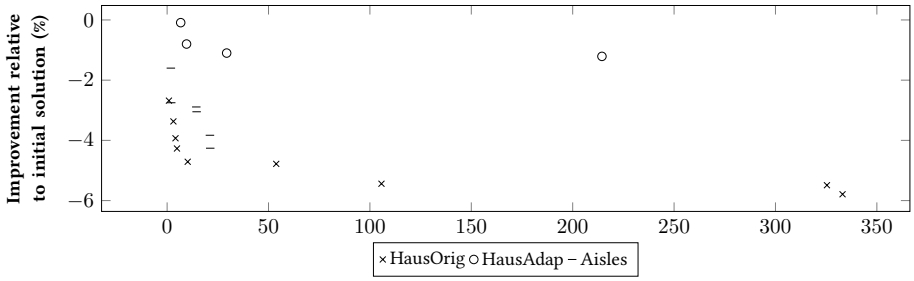


Figure 6: Evolution in time of the solution's objective for instance *40s-60-75-0*, solved by the original 2level-VNS algorithm with the HausOrig, HausAdap or Aisles heuristic implemented. Each mark refers to the moment when a better solution is found, with on the y-axis the % improvement relative to the initial solution found by the respective batching heuristic.

the initial solution found by the respective batching heuristic. For all batching heuristics tested, improvements to the solution are significant during the first minute. After 60 seconds, the solution improves only little, while the time between improvements generally increases. For further experiments we decide to terminate the 2level-VNS after 60 seconds. We denote the 2level-VNS algorithm with time-based stopping criterion as *2level-VNS60*.

6.3 Two-level VNS approach - Comparison of batching heuristics

In this section, we analyse the results for the HausOrig, HausAdap and Aisles heuristic, to conclude which one performs best when implemented in the 2level-VNS60 algorithm. In section 6.3.1, we compare the performance of the batching heuristics in a pairwise manner for the Ran_32 dataset. To validate our observations, we perform similar experiments on a larger dataset. Results are provided and discussed in section 6.3.2.

6.3.1 Results for limited dataset, Ran_32

For the Ran_32 instances, we analyse the results of the HausOrig, HausAdap and Aisles batching heuristic, summarized in table 14. We compare the heuristics in a pairwise manner and count the number of instances for which each was able to find the *best solution*, i.e., the best of both outcomes, over the set. This number includes instances for which both batching heuristics obtained the same total travel distance. For instances for which the heuristic performs worse, we compute the average deviation relative to the best solution. Detailed results are provided in Appendix A.
Of both Hausdorff-based batching criteria, the HausAdap clearly finds the best solution for more instances than the HausOrig heuristic. For the remaining instances, the solutions provided by HausAdap are only 0.14% worse, in contrast to 0.46% when the HausOrig heuristic is adopted. Despite the large similarities between the CluVRP and

JOBPRP, these findings prove the merit of our adaptations to the original Hausdorff batching heuristic in order to produce JOBPRP-fit solutions.
Despite these adaptations, we observe that the minimal aisles visited-criterion still performs better. The Aisles heuristic is able to find the best solution for 62.5% of the instances. For the remaining instances, the aisle-based heuristic deviates on average 0.38% from the best solution, while a slightly larger average is recorded for the HausAdap heuristic (0.54%). Before stating a clear preference, for any batching heuristic, we validate our observations with a similar analysis conducted on a larger dataset. Results are discussed in the next section.

		# best solution			**Avg. solution gap (%)**			**Computatation time (sec)**		
	Nb. Inst.	Haus Orig	Haus Adap	Aisles	Haus Orig	Haus Adap	Aisles	Haus Orig	Haus Adap	Aisles
2level-VNS60										
Ran_32								60	60	60
HausOrig-HausAdap	32	13	20		0.46%	0.14%				
HausOrig-Aisles	32	9		23	0.75%		0.45%			
HausAdap-Aisles	32		12	20		0.54%	0.38%			

Table 14: Pairwise comparison of the HausOrig, HausAdap and Aisles batching heuristic when implemented in the 2level-VNS60 algorithm, applied to the instances of the Ran_32 dataset.

6.3.2 Validation on larger dataset, Ran_1280

We extend the pairwise comparisons to a larger dataset, that is the initial set from which Ran_32 was drawn. Each (n,C)-combination (values given in table 12) is now represented by 80 instances instead of two, leading to a set of 1280 instances, referred to as *Ran_1280*.
We repeat the experiments of section 6.3.1 on the instances of Ran_1280, and record the results in table 15. Based on this set, which statistically includes more variation than Ran_32, we confirm that the HausAdap implementation outperforms the HausOrig heuristic. Moreover, we find for this larger dataset more obvious preference for the Aisles heuristic in comparison to the best performing Hausdorff based-heuristic. The aisle-based heuristic is able to find a better (or the same) solution for a double amount of instances than the HausAdap heuristic.

		# best solutions			**Avg. gap (%)**			**Computation time (sec)**		
	Nb. Inst.	Haus Orig	Haus Adap	Aisles	Haus Orig	Haus Adap	Aisles	Haus Orig	Haus Adap	Aisles
2level-VNS60										
Ran_1280								60	60	60
HausOrig-HausAdap	1280	454	840		0.51%	0.20%				
HausOrig-Aisles	1280	298		990	0.85%		0.35%			
HausAdap-Aisles	1280		412	879		0.63%	0.36%			

Table 15: Pairwise comparison of the HausOrig, HausAdap and Aisles batching heuristic implemented in the 2level-VNS60 algorithm, applied to the instances of the Ran_1280 dataset.

Notwithstanding the results in favour of the Aisles heuristic, we remark that a small subset of instances (32.5%) seems to perform better when the HausAdap heuristic

is implemented. By decomposing the results by number of orders and batch capacity, we find that especially the latter influences these results. This decomposition is graphically presented in table 16. Each cell summarizes the results for one (n,C)-combination, represented by 80 instances. The bars show the percentage of instances for which the HausAdap, respectively Aisles heuristic is able to find the best solution.

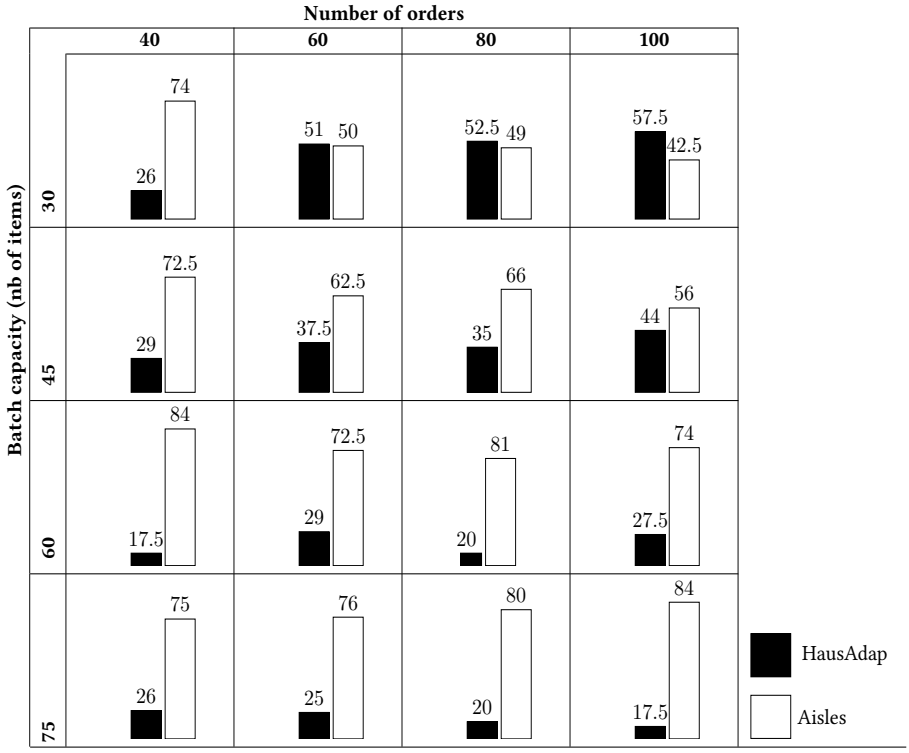


Table 16: Pairwise comparison of the HausAdap and Aisles heuristic, broken down by number of orders and batch capacity. Each cell represents 80 instances.

When the batch capacity is set at 30 items, we observe that the HausAdap heuristic often performs better than the Aisles heuristic, except when the number of orders is rather small. However, the percentages remain too low to declare a clear preference for the HausAdap heuristic for this particular set of instances. As the batch capacity grows, the superiority of the Aisles heuristic increases.
We conclude that the results found for the Ran_32 subset align with the results obtained for the extended dataset, Ran_1280. We are able to confirm that the HausOrig heuristic is outperformed by both the HausAdap and Aisles heuristic. Regarding the comparison of the HausAdap and Aisles heuristic, we found evidence for the Ran_32 instances to state a preference for the latter, which was validated by the Ran_1280 dataset. In particular, there is a larger absolute number of instances for which the Aisles heuristic performs better than the HausAdap heuristic. However, the dominance of the Aisles heuristic is only visible for 69% of the instances and we decide to

continue experiments with both the HausAdap and Aisles heuristics.

6.4 Two-level VNS approach - Comparison with state of the art algorithms

In this section, we compare the 2level-VNS60 algorithm with the following state-of-the-art OBP algorithms:

- Variable Neighborhood Descent (VND) approach by Albareda-Sambola et al. (2009)
- Attribute-Based Hill Climbing approach with s-shape routing (AHBC + SS) by Henn and Wäscher (2012)
- Attribute-Based Hill Climbing approach with largest gap routing (AHBC + LG) by Henn and Wäscher (2012)
- Multi-Start Variable Neighborhood Search (MS-VNS) by Menéndez et al. (2017)

Following analyses are based on the results for the Ran_32 set, made available by Menéndez et al. (2017) (available at `http://grafo.etsii.urjc.es/optsicom/obp/`(Menéndez et al., 2019))[1].
In table 17 we compare the 2level-VNS60 and the four algorithms mentioned above. This comparison is performed twice: once when implementing the HausAdap heuristic in the 2level-VNS60, once with the Aisles heuristic adopted. The results are found in the penultimate, respectively last column. We start with an overall comparison for which we report how many times each algorithm is able to find the best solution out of the five obtained results. Next, we conduct a pairwise comparison between the 2level-VNS60 and each of the four state-of-the-art algorithms. We report how many times the 2level-VNS60 is able to find the best result, as well as the average improvement (%) achieved by the algorithm for that particular set of instances. For the cases where the 2level-VNS60 performed worse, we report the average deviation (%) relative to the best solution. We close table 17 with the average computation time (in seconds).
Overall, the 2level-VNS60 performs well for both the HausAdap and Aisles batching heuristic. It outperforms the VND and ABHC+LG algorithms for all instances, with an average improvement up to 6.20 and 10.53%, respectively. Compared to the ABHC+SS method, the 2level-VNS60 performs better for 22 out of the 32 instances, and improves the ABHC+SS solution up to 6.29% on average for those 22 cases. Results are similar for the HausAdap and Aisles heuristic, although in general, larger improvements are obtained with the Aisles heuristics. Differences, however, are minor.
In comparison to the MS-VNS, the 2level-VNS60 performs better for only 8 out of 32 instances. From the detailed results provided in Appendix A, we derive that, independent of the implemented batching heuristic, the 2level-VNS60 often finds better

[1]The results of the 2level-VNS are compared to publicly available results obtained on an Intel QuadCore with 2.5 GHz and 6 GB of RAM with Xubuntu 14.04 64 bit OS, a platform that is less performant than ours. However, the intention of this paper is not to compare the solution quality of our algorithm in terms of CPU performance, but in terms of total picker travel distance.

solutions than the MS-VNS method for instances where the batch capacity is small (30 or 45 items). For larger batch size ($\geq$ 60 items) the MS-VNS method consistently performs better.

		Ran_32		
	# inst.	**Algorithm**	**2level-VNS60 HausAdap**	**2level-VNS60 Aisles**
Overall comparison				
# best solutions	32	**2level-VNS60**	8	8
		VND	0	0
		ABHC+SS	7	7
		ABHC+LG	0	0
		MS-VNS	17	17
Pairwise comparison				
# best solutions by 2level-VNS60	32	VND	32	32
		ABHC+SS	22	22
		ABHC+LG	32	32
		MS-VNS	8	8
Avg. improvement (%)		VND	-6.02%	-6.20%
when 2level-VNS60 better		ABHC+SS	-6.14%	-6.29%
		ABHC+LG	-10.37%	-10.53%
		MS-VNS	-1.51%	-1.38%
Avg. gap (%) 2level-VNS60		VND	-	-
in respect of best found solution		ABHC+SS	2.88%	2.60%
		ABHC+LG	-	-
		MS-VNS	1.81%	1.51%
Avg. computation time (sec)		2level VNS60	60	60
		VND	0.66	0.66
		ABHC+SS	13.45	13.45
		ABHC+LG	61.57	61.57
		MS-VNS	49.09	49.09

Table 17: Comparison of the 2level-VNS60 with state-of-the-art OBP algorithms. Table shows the number of times the 2level-VNS60 or OBP algorithms obtain the best solution for the Ran_32 instances, with respectively the HausAdap or Aisles batching heuristic implemented.

Finally, we perform a similar comparison with the original 2level-VNS algorithm, i.e., the 2level-VNS that terminates after 1000 consecutive iterations without improvement. Results are reported in table 18, which deviate only little from the results in table 17. This is remarkable, given that the average computation time required by the original 2level-VNS is significantly larger than the 60 seconds adopted in the 2level-VNS60. Overall, we find prior conclusions to remain valid and confirm that the adoption of a time-based criterion is a good approach to retain performant solutions, competitive with state of the art OBP algorithms.
We conclude that the 2level-VNS60 algorithm is able to outperform the VND, ABHC+SS and ABHC+LG algorithms for the majority of instances. However, our algorithm is not able to dominate the MS-VNS approach, although the solutions deviate to a minor extent ($<$1.81%). Concerning the batching criteria, we conclude that when the 2level-VNS60 approach is used, it is best to implement the aisle-based batching heuristic.

		Ran_32		
	# inst.	**Algorithm**	**2level-VNS HausAdap**	**2level-VNS Aisles**
Overall comparison				
# best solution	32	**2level-VNS**	9	11
		VND	0	0
		ABHC+SS	7	7
		ABHC+LG	0	0
		MS-VNS	16	14
Pairwise comparison				
# best solution by 2level-VNS		VND	32	32
		ABHC+SS	22	23
		ABHC+LG	32	32
		MS-VNS	10	11
Avg. improvement (%)		VND	-6.41%	-6.61%
when 2level VNS better		ABHC+SS	-6.50%	-6.38%
		ABHC+LG	-10.74%	-10.93%
		MS-VNS	-1.36%	-1.22%
Avg. gap (%) 2level-VNS		VND	-	-
in respect of best found solution		ABHC+SS	2.36%	2.30%
		ABHC+LG	-	-
		MS-VNS	1.43%	1.16%
Avg. computation time (sec)		2level VNS	576.51	486.07
		VND	0.66	0.66
		ABHC+SS	13.45	13.45
		ABHC+LG	61.57	61.57
		MS-VNS	49.09	49.09

Table 18: Comparison of the 2level-VNS including original stopping criterion with state-of-the-art OBP algorithms. Table shows the number of times the 2level-VNS or OBP algorithms obtain the best solution for the Ran_32 instances, with respectively the HausAdap or Aisles batching heuristic implemented.

7 Conclusion

We demonstrate that the joint order batching and picker routing problem (JOBPRP), studied in the warehouse literature, and the clustered vehicle routing problem (CluVRP), studied in the VRP literature, share many similarities. Despite the mathematical overlap of both problems, only limited attempts have been found to use existing CluVRP algorithms when solving the JOBPRP. In this paper, we propose the two-level VNS presented by Defryn and Sörensen (2017) for the CluVRP, as a metaheuristic approach to solve the JOBPRP. In contrast to existing algorithms for the (J)OBP(RP), our approach does not primarily assign orders to batches based on the total travel distance, but rather utilizes the Hausdorff distance between orders to compose batches in a first phase. We implemented the Hausdorff-based batching heuristic (HausOrig) as originally proposed by Defryn and Sörensen (2017), and proposed an adapted version (HausAdap) to create a better alignment with the JOBPRP. Both the HausOrig and HausAdap heuristics were compared with an aisle-based batching heuristic, regularly used in warehouse literature.

We consider the 2level-VNS with adapted Hausdorff batching heuristic to be successful as it found a better solution for almost twice the number of instances than the original Hausdorff-based batching heuristic. Nonetheless, we observe that the aisle-based heuristic remains superior. The Aisles heuristic finds the best solution for 69% of the instance set, in contrast to 32.5% when the adapted Hausdorff batching heuristic is implemented. Moreover, we find that if the Aisles heuristic performs worse, the average deviation relative to the best solution remains small. We therefore conclude a preference for the aisle-based batching heuristic. Additionally, we compared the 2levelVNS to state of the art OBP algorithms. Our 2level-VNS, modified to a time-based stopping criterion for a fair comparison, outperformed three of the four algorithms, with average improvements between 6% and 10%. However, the MS-VNS algorithm, developed by Menéndez et al. (2017), maintains its superiority and performs best, especially when the batch capacity is large.

We note that these conclusions hold for a single-block warehouse with parallel aisles and the instance characteristics used in this study. We acknowledge the need of further studies on other warehouse layouts and other instance sets to validate the conclusions regarding the performance of the Hausdorff heuristic and proposed metaheuristic in solving the JOBPRP.

In this paper, we showed that more similarities can be found between the vehicle routing context and warehouse environment besides the resemblances between the TSP and PRP. We therefore encourage future experimentation of existing VRP algorithms for warehousing problems that share similar structures with their VRP-counterparts although, we do recommend to implement small adaptations to ensure a proper fit with the problem under study.

Acknowledgements

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Appendix A. Detailed results

Appendix A shows the detailed results when comparing the original 2level-VNS algorithm (table A.1), respectively 2level-VNS60 algorithm (table A.2) to state-of-the-art OBP algorithms and is complimentary to the data provided in section 6.4.

			Total travel distance (m)	Avg. deviation (%) relative to outcome OBP algorithm			
	n	C	2level-VNS	VND	ABHC+SS	ABHC+LG	MS-VNS
HausOrig	40	30	9390	-6.82%	-6.40%	-6.54%	-1.87%
	40	30	9826	-6.77%	-12.00%	-9.47%	-1.94%
	40	45	6239	-6.78%	-1.52%	-11.29%	-0.06%
	40	45	7257	-6.61%	-8.06%	-11.99%	-1.36%
	40	60	5674	-7.93%	2.75%	-13.10%	0.76%
	40	60	5034	-6.41%	-5.41%	-12.70%	0.72%
	40	75	4433	-7.08%	2.15%	-15.09%	1.09%
	40	75	4071	-4.53%	-5.76%	-17.32%	1.07%
	60	30	14825	-7.16%	-5.77%	-8.31%	-0.90%
	60	30	13660	-9.23%	-13.87%	-10.27%	-2.49%
	60	45	9628	-7.02%	0.15%	-6.72%	0.02%
	60	45	10458	-6.20%	-4.69%	-10.19%	1.78%
	60	60	7844	-7.28%	0.40%	-10.15%	1.08%
	60	60	7448	-5.92%	-4.21%	-12.73%	1.71%
	60	75	5562	-6.60%	1.48%	-14.05%	1.63%
	60	75	5590	-3.97%	-3.04%	-14.97%	0.88%
	80	30	19223	-7.23%	-9.76%	-5.64%	-2.31%
	80	30	17533	-6.66%	-14.10%	-9.16%	-0.14%
	80	45	14204	-5.93%	-0.83%	-6.89%	1.44%
	80	45	12045	-6.95%	-5.94%	-10.27%	1.40%
	80	60	9990	-3.49%	3.15%	-5.81%	2.11%
	80	60	9977	-7.43%	-4.14%	-12.00%	1.74%
	80	75	8149	-6.56%	3.88%	-11.65%	1.74%
	80	75	8199	-5.67%	-3.13%	-15.30%	1.52%
	100	30	21017	-8.36%	-7.99%	-5.78%	-0.52%
	100	30	23068	-8.31%	-12.63%	-9.58%	-1.97%
	100	45	14756	-3.42%	1.21%	-4.49%	2.29%
	100	45	14664	-5.33%	-5.87%	-9.78%	1.67%
	100	60	11796	-4.26%	5.34%	-7.84%	3.32%
	100	60	13434	-4.87%	-4.62%	-12.29%	1.75%
	100	75	9635	-5.41%	5.55%	-14.76%	2.55%
	100	75	9970	-4.25%	-0.61%	-13.17%	4.37%
HausAdap	40	30	9395	-6.77%	-6.35%	-6.49%	-1.82%
	40	30	9821	-6.81%	-12.05%	-9.52%	-1.99%
	40	45	6249	-6.63%	-1.37%	-11.15%	0.10%
	40	45	7263	-6.54%	-7.98%	-11.92%	-1.28%
	40	60	5645	-8.40%	2.23%	-13.54%	0.25%
	40	60	5010	-6.86%	-5.86%	-13.11%	0.24%
	40	75	4431	-7.13%	2.11%	-15.13%	1.05%
	40	75	4078	-4.36%	-5.60%	-17.18%	1.24%
	60	30	14817	-7.21%	-5.83%	-8.36%	-0.96%
	60	30	13673	-9.14%	-13.78%	-10.19%	-2.40%
	60	45	9626	-7.04%	0.12%	-6.74%	0.00%
	60	45	10322	-7.42%	-5.93%	-11.35%	0.46%
	60	60	7824	-7.52%	0.14%	-10.38%	0.82%
	60	60	7442	-6.00%	-4.28%	-12.80%	1.63%
	60	75	5560	-6.63%	1.44%	-14.08%	1.59%
	60	75	5583	-4.09%	-3.16%	-15.07%	0.76%
	80	30	19229	-7.20%	-9.73%	-5.61%	-2.28%
	80	30	17532	-6.67%	-14.10%	-9.16%	-0.15%
	80	45	14223	-5.80%	-0.70%	-6.76%	1.57%
	80	45	12072	-6.74%	-5.72%	-10.07%	1.62%
	80	60	9952	-3.85%	2.76%	-6.17%	1.72%
	80	60	9979	-7.41%	-4.12%	-11.99%	1.76%

(continued on next page)

Table 19 - *Continued*

			Total travel distance (m)	Avg. deviation (%) relative to outcome OBP algorithm			
	n	C	2level-VNS	VND	ABHC+SS	ABHC+LG	MS-VNS
	80	75	8129	-6.79%	3.62%	-11.87%	1.49%
	80	75	8212	-5.52%	-2.98%	-15.17%	1.68%
	100	30	20914	-8.81%	-8.44%	-6.24%	-1.01%
	100	30	23125	-8.08%	-12.41%	-9.36%	-1.73%
	100	45	14785	-3.23%	1.41%	-4.30%	2.50%
	100	45	14645	-5.46%	-6.00%	-9.90%	1.54%
	100	60	11693	-5.10%	4.42%	-8.65%	2.42%
	100	60	13430	-4.90%	-4.65%	-12.31%	1.72%
	100	75	9618	-5.58%	5.37%	-14.91%	2.37%
	100	75	9832	-5.57%	-1.98%	-14.37%	2.92%
Aisles	40	30	9380	-6.92%	-6.50%	-6.64%	-1.98%
	40	30	9820	-6.82%	-12.05%	-9.53%	-2.00%
	40	45	6251	-6.60%	-1.33%	-11.12%	0.13%
	40	45	7208	-7.24%	-8.68%	-12.59%	-2.03%
	40	60	5645	-8.40%	2.23%	-13.54%	0.25%
	40	60	4992	-7.19%	-6.20%	-13.42%	-0.12%
	40	75	4446	-6.81%	2.45%	-14.84%	1.39%
	40	75	4053	-4.95%	-6.18%	-17.69%	0.62%
	60	30	14874	-6.85%	-5.46%	-8.01%	-0.57%
	60	30	13660	-9.23%	-13.87%	-10.27%	-2.49%
	60	45	9630	-7.00%	0.17%	-6.70%	0.04%
	60	45	10242	-8.14%	-6.66%	-12.04%	-0.32%
	60	60	7843	-7.29%	0.38%	-10.16%	1.07%
	60	60	7417	-6.32%	-4.60%	-13.09%	1.28%
	60	75	5476	-8.04%	-0.09%	-15.38%	0.05%
	60	75	5531	-4.98%	-4.06%	-15.87%	-0.18%
	80	30	19262	-7.05%	-9.58%	-5.44%	-2.11%
	80	30	17615	-6.22%	-13.69%	-8.73%	0.32%
	80	45	14191	-6.01%	-0.93%	-6.97%	1.34%
	80	45	12011	-7.22%	-6.20%	-10.53%	1.11%
	80	60	9988	-3.51%	3.13%	-5.83%	2.09%
	80	60	9924	-7.92%	-4.65%	-12.47%	1.20%
	80	75	8092	-7.21%	3.15%	-12.27%	1.02%
	80	75	8193	-5.74%	-3.20%	-15.36%	1.45%
	100	30	21048	-8.22%	-7.85%	-5.64%	-0.37%
	100	30	23240	-7.63%	-11.98%	-8.91%	-1.24%
	100	45	14697	-3.80%	0.81%	-4.87%	1.89%
	100	45	14653	-5.40%	-5.94%	-9.85%	1.59%
	100	60	11566	-6.13%	3.29%	-9.64%	1.31%
	100	60	13469	-4.62%	-4.37%	-12.06%	2.01%
	100	75	9591	-5.84%	5.07%	-15.15%	2.09%
	100	75	9757	-6.29%	-2.73%	-15.02%	2,14%

Table A.1: Detailed results for the original 2level-VNS algorithm with stopping criterion '1000 consecutive iterations without improvement' with HausOrig, HausAdap or Aisles heuristic implemented. Results are provided for the Ran_32 instances.

			Total travel distance (m)	Avg. deviation (%) relative to outcome OBP algorithm			
	n	C	2level-VNS60	VND	ABHC+SS	ABHC+LG	MS-VNS
HausAdap	40	30	9399	-6.73%	-6.31%	-6.45%	-1.78%
	40	30	9833	-6.70%	-11.94%	-9.41%	-1.87%
	40	45	6269	-6.33%	-1.05%	-10.86%	0.42%
	40	45	7265	-6.51%	-7.96%	-11.90%	-1.25%
	40	60	5680	-7.84%	2.86%	-13.00%	0.87%
	40	60	5025	-6.58%	-5.58%	-12.85%	0.54%
	40	75	4454	-6.64%	2.64%	-14.69%	1.57%
	40	75	4086	-4.17%	-5.42%	-17.02%	1.44%
	60	30	14840	-7.06%	-5.68%	-8.22%	-0.80%
	60	30	13677	-9.12%	-13.76%	-10.16%	-2.37%
	60	45	9669	-6.62%	0.57%	-6.33%	0.45%
	60	45	10370	-6.99%	-5.50%	-10.94%	0.92%
	60	60	7901	-6.61%	1.13%	-9.50%	1.82%
	60	60	7472	-5.62%	-3.90%	-12.44%	2.03%
	60	75	5550	-6.80%	1.26%	-14.23%	1.41%
	60	75	5640	-3.11%	-2.17%	-14.21%	1.79%
	80	30	19292	-6.90%	-9.44%	-5.30%	-1.96%
	80	30	17587	-6.37%	-13.83%	-8.88%	0.17%
	80	45	14224	-5.80%	-0.69%	-6.76%	1.58%
	80	45	12126	-6.33%	-5.30%	-9.67%	2.08%
	80	60	9982	-3.56%	3.07%	-5.88%	2.02%
	80	60	10062	-6.64%	-3.32%	-11.25%	2.61%
	80	75	8195	-6.03%	4.46%	-11.16%	2.31%
	80	75	8271	-4.84%	-2.28%	-14.56%	2.41%
	100	30	21103	-7.98%	-7.61%	-5.39%	-0.11%
	100	30	23348	-7.20%	-11.57%	-8.48%	-0.78%
	100	45	14799	-3.14%	1.51%	-4.21%	2.59%
	100	45	14725	-4.94%	-5.48%	-9.41%	2.09%
	100	60	11623	-5.67%	3.80%	-9.20%	1.80%
	100	60	13546	-4.08%	-3.83%	-11.56%	2.60%
	100	75	9609	-5.66%	5.27%	-14.99%	2.28%
	100	75	9801	-5.87%	-2.29%	-14.64%	2.60%
Aisles	40	30	9371	-7.01%	-6.59%	-6.73%	-2.07%
	40	30	9809	-6.93%	-12.15%	-9.63%	-2.11%
	40	45	6277	-6.22%	-0.92%	-10.75%	0.54%
	40	45	7234	-6.91%	-8.35%	-12.27%	-1.67%
	40	60	5634	-8.58%	2.03%	-13.71%	0.05%
	40	60	5007	-6.92%	-5.92%	-13.16%	0.18%
	40	75	4485	-5.99%	3.35%	-14.10%	2.28%
	40	75	4062	-4.74%	-5.97%	-17.51%	0.84%
	60	30	14938	-6.45%	-5.06%	-7.61%	-0.15%
	60	30	13696	-8.99%	-13.64%	-10.04%	-2.23%
	60	45	9675	-6.57%	0.63%	-6.27%	0.51%
	60	45	10318	-7.45%	-5.97%	-11.39%	0.42%
	60	60	7897	-6.65%	1.08%	-9.54%	1.77%
	60	60	7446	-5.95%	-4.23%	-12.75%	1.68%
	60	75	5517	-7.36%	0.66%	-14.74%	0.80%
	60	75	5553	-4.60%	-3.68%	-15.53%	0.22%
	80	30	19302	-6.85%	-9.39%	-5.25%	-1.91%
	80	30	17713	-5.70%	-13.21%	-8.22%	0.88%
	80	45	14259	-5.56%	-0.45%	-6.53%	1.83%
	80	45	12044	-6.96%	-5.94%	-10.28%	1.39%
	80	60	10050	-2.91%	3.77%	-5.24%	2.72%
	80	60	10035	-6.89%	-3.58%	-11.49%	2.34%

(continued on next page)

Table 20 - *Continued*

		Total travel distance (m)	Avg. deviation (%) relative to outcome OBP algorithm			
n	C	2level-VNS60	VND	ABHC+SS	ABHC+LG	MS-VNS
80	75	8151	-6.54%	3.90%	-11.63%	1.76%
80	75	8240	-5.20%	-2.65%	-14.88%	2.03%
100	30	21103	-7.98%	-7.61%	-5.39%	-0.11%
100	30	23348	-7.20%	-11.57%	-8.48%	-0.78%
100	45	14799	-3.14%	1.51%	-4.21%	2.59%
100	45	14725	-4.94%	-5.48%	-9.41%	2.09%
100	60	11623	-5.67%	3.80%	-9.20%	1.80%
100	60	13546	-4.08%	-3.83%	-11.56%	2.60%
100	75	9609	-5.66%	5.27%	-14.99%	2.28%
100	75	9801	-5.87%	-2.29%	-14.64%	2.60%

Table A.2: Detailed results for the 2level-VNS60 algorithm, with HausAdap or Aisles heuristic implemented. Results are shown for the Ran_32 instances.

References

Albareda-Sambola, M., Alonso-Ayuso, A., Molina, E., and De Blas, C. S. (2009). Variable neighborhood search for order batching in a warehouse. *Asia-Pacific Journal of Operational Research*, 26(05):655–683.

Barthélemy, T., Rossi, A., Sevaux, M., and Sörensen, K. (2010). Metaheuristic approach for the clustered vrp. In *EU/Meeting: 10th Anniversary of the Metaheuristics Community-Université de Bretagne Sud, France*.

Battarra, M., Erdoğan, G., and Vigo, D. (2014). Exact algorithms for the clustered vehicle routing problem. *Operations Research*, 62(1):58–71.

Bektaş, T., Erdoğan, G., and Røpke, S. (2011). Formulations and branch-and-cut algorithms for the generalized vehicle routing problem. *Transportation Science*, 45(3):299–316.

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

Cambazard, H. and Catusse, N. (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.

Cheng, C.-Y., Chen, Y.-Y., Chen, T.-L., and Yoo, J. J.-W. (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.

Chisman, J. A. (1975). The clustered traveling salesman problem. *Computers & Operations Research*, 2(2):115–119.

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

De Koster, R., Van der Poort, E., and Wolters, M. (1999). Efficient orderbatching methods in warehouses. *International Journal of Production Research*, 37(7):1479–1504.

Defryn, C. and Sörensen, K. (2015). A two-level variable neighbourhood search for the euclidean clustered vehicle routing problem. University of Antwerp, Faculty of Applied Economics Research Paper, 2015-002. Available at `https://repository.uantwerpen.be/docman/irua/295901/a0dff71b.pdf`.

Defryn, C. and Sörensen, K. (2017). A fast two-level variable neighborhood search for the clustered vehicle routing problem. *Computers & Operations Research*, 83:78–94.

Ding, C., Cheng, Y., and He, M. (2007). Two-level genetic algorithm for clustered traveling salesman problem with application in large-scale tsps. *Tsinghua Science & Technology*, 12(4):459–465.

Expósito-Izquierdo, C., Rossi, A., and Sevaux, M. (2016). A two-level solution approach to solve the clustered capacitated vehicle routing problem. *Computers & Industrial Engineering*, 91:274–289.

Gademann, N. and Velde, S. (2005). Order batching to minimize total travel time in a parallel-aisle warehouse. *IIE transactions*, 37(1):63–75.

Ghiani, G. and Improta, G. (2000). An efficient transformation of the generalized vehicle routing problem. *European Journal of Operational Research*, 122(1):11–17.

Gibson, D. R. and Sharp, G. P. (1992). Order batching procedures. *European Journal of Operational Research*, 58(1):57–67.

Henn, S. and Wäscher, G. (2012). Tabu search heuristics for the order batching problem in manual order picking systems. *European Journal of Operational Research*, 222(3):484–494.

Hintsch, T. et al. (2019). Large multiple neighborhood search for the soft-clustered vehicle-routing problem. Technical report.

Hintsch, T. and Irnich, S. (2018). Large multiple neighborhood search for the clustered vehicle-routing problem. *European Journal of Operational Research*, 270(1):118–131.

Hintsch, T. and Irnich, S. (2020). Exact solution of the soft-clustered vehicle-routing problem. *European Journal of Operational Research*, 280(1):164–178.

Ho, Y.-C., Su, T.-S., and Shi, Z.-B. (2008). Order-batching methods for an order-picking warehouse with two cross aisles. *Computers & Industrial Engineering*, 55(2):321–347.

Ho, Y.-C. and Tseng, Y.-Y. (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.

Horvat-Marc, A., Fuksz, L., Pop, P. C., and Dănciulescu, D. (2018). A decomposition-based method for solving the clustered vehicle routing problem. *Logic Journal of the IGPL*, 26(1):83–95.

Hsu, C.-M., Chen, K.-Y., and Chen, M.-C. (2005). Batching orders in warehouses by minimizing travel distance with genetic algorithms. *Computers in Industry*, 56(2):169–178.

Hung, W.-L. and Yang, M.-S. (2004). Similarity measures of intuitionistic fuzzy sets based on hausdorff distance. *Pattern Recognition Letters*, 25(14):1603–1611.

Kulak, O., Sahin, Y., and Taner, M. (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(1):52–80.

Laporte, G., Potvin, J.-Y., and Quilleret, F. (1997). A tabu search heuristic using genetic diversification for the clustered traveling salesman problem. *Journal of Heuristics*, 2(3):187–200.

Löffler, M., Boysen, N., and Schneider, M. (2018). Picker routing in agv-assisted order picking systems. Technical report, Deutsche Post Chair-Optimization of Distribution Working Paper, DPO-01/2018,.

Marc, A. H., Fuksz, L., Pop, P. C., and Dănciulescu, D. (2015). A novel hybrid algorithm for solving the clustered vehicle routing problem. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 679–689. Springer.

Menéndez, B., Pardo, E. G., Alonso-Ayuso, A., Molina, E., and Duarte, A. (2016 (accessed August 5, 2019)). Order batching problem, instances. http://grafo.etsii.urjc.es/optsicom/obp/.

Menéndez, B., Pardo, E. G., Alonso-Ayuso, A., Molina, E., and Duarte, A. (2017). Variable neighborhood search strategies for the order batching problem. *Computers & Operations Research*, 78:500–512.

Öncan, T. (2015). Milp formulations and an iterated local search algorithm with tabu thresholding for the order batching problem. *European Journal of Operational Research*, 243(1):142–155.

Petersen, C. G. and Schmenner, R. W. (1999). An evaluation of routing and volume-based storage policies in an order picking operation. *Decision Sciences*, 30(2):481–501.

Pop, P. and Chira, C. (2014). A hybrid approach based on genetic algorithms for solving the clustered vehicle routing problem. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 1421–1426. IEEE.

Pop, P. C., Fuksz, L., Marc, A. H., and Sabo, C. (2018). A novel two-level optimization approach for clustered vehicle routing problem. *Computers & Industrial Engineering*, 115:304–318.

Pop, P. C., Kara, I., and Marc, A. H. (2012). New mathematical models of the generalized vehicle routing problem and extensions. *Applied Mathematical Modelling*, 36(1):97–107.

Pop, P. C., Matei, O., and Sitar, C. P. (2013). An improved hybrid algorithm for solving the generalized vehicle routing problem. *Neurocomputing*, 109:76–83.

Pop, P. C., Zelina, I., Lupşe, V., Sitar, C. P., and Chira, C. (2011). Heuristic algorithms for solving the generalized vehicle routing problem. *International Journal of Computers Communications & Control*, 6(1):158–165.

Posada, A., Rivera, J. C., and Palacio, J. D. (2018). A mixed-integer linear programming model for a selective vehicle routing problem. In *Workshop on Engineering Applications*, pages 108–119. Springer.

Posada, A., Rivera, J. C., and Palacio, J. D. (2019). Selective vehicle routing problem: A hybrid genetic algorithm approach. In *International Conference on Artificial Evolution (Evolution Artificielle)*, pages 148–161. Springer.

Ratliff, H. D. and Rosenthal, A. S. (1983). Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem. *Operations Research*, 31(3):507–521.

Roodbergen, K.-J. (2001). *Layout and routing methods for warehouses*. PhD thesis, Erasmus Universiteit Rotterdam.

Roodbergen, K. J. and Koster, R. (2001). Routing methods for warehouses with multiple cross aisles. *International Journal of Production Research*, 39(9):1865–1883.

Rosenwein, M. (1996). A comparison of heuristics for the problem of batching orders for warehouse selection. *International Journal of Production Research*, 34(3):657–664.

Sabo, C., Pop, P. C., and Horvat-Marc, A. (2020). On the selective vehicle routing problem. *Mathematics*, 8(5):771.

Scholz, A. and Wäscher, G. (2017). Order batching and picker routing in manual order picking systems: the benefits of integrated routing. *Central European Journal of Operations Research*, 25(2):491–520.

Sevaux, M., Sörensen, K., et al. (2008). Hamiltonian paths in large clustered routing problems. In *Proceedings of the EU/MEeting 2008 workshop on Metaheuristics for Logistics and Vehicle Routing, EU/ME*, volume 8, pages 411–417.

Sim, D.-G., Kwon, O.-K., and Park, R.-H. (1999). Object matching algorithms using robust hausdorff distance measures. *IEEE Transactions on image processing*, 8(3):425–429.

Theys, C., Bräysy, O., Dullaert, W., and Raa, B. (2010). Using a tsp heuristic for routing order pickers in warehouses. *European Journal of Operational Research*, 200(3):755–763.

Tsai, C.-Y., Liou, J., and Huang, T.-M. (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, C. A., Beasley, J. E., and da Cunha, A. S. (2017). Optimally solving the joint order batching and picker routing problem. *European Journal of Operational Research*, 262(3):817–834.

Van Gils, T., Ramaekers, K., Braekers, K., Depaire, B., and Caris, A. (2018). Increasing order picking efficiency by integrating storage, batching, zone picking, and routing policy decisions. *International Journal of Production Economics*, 197:243–261.

Vidal, T., Battarra, M., Subramanian, A., and Erdogan, G. (2015). Hybrid metaheuristics for the clustered vehicle routing problem. *Computers & Operations Research*, 58:87–99.

Won, J. and Olafsson, S. (2005). Joint order batching and order picking in warehouse operations. *International Journal of Production Research*, 43(7):1427–1442.

Yu, M. and De Koster, R. B. (2009). The impact of order batching and picking area zoning on order picking system performance. *European Journal of Operational Research*, 198(2):480–490.

Zhang, J., Wang, X., Chan, F. T., and Ruan, J. (2017). On-line order batching and sequencing problem with multiple pickers: A hybrid rule-based algorithm. *Applied Mathematical Modelling*, 45:271–284.