DEPARTMENT OF ENGINEERING MANAGEMENT

Solving the joint order batching and picker routing problem, as a clustered vehicle routing problem

Babiche Aerts, Trijntje Cornelissens & Kenneth Sörensen

UNIVERSITY OF ANTWERP
Faculty of Business and Economics
City Campus
Prinsstraat 13, B.226
B-2000 Antwerp
Tel. +32 (0)3 265 40 32
www.uantwerpen.be

FACULTY OF BUSINESS AND ECONOMICS

DEPARTMENT OF ENGINEERING MANAGEMENT

Solving the joint order batching and picker routing problem, as a clustered vehicle routing problem

Babiche Aerts, Trijntje Cornelissens & Kenneth Sörensen

RESEARCH PAPER 2020-003
APRIL 2020

University of Antwerp, City Campus, Prinsstraat 13, B-2000 Antwerp, Belgium
Research Administration – room B.226
phone: (32) 3 265 40 32
e-mail: joeri.nys@uantwerpen.be

The research papers from the Faculty of Business and Economics are also available at www.repec.org (Research Papers in Economics - RePEc)

D/2020/1169/003

Solving the joint order batching and picker routing problem, as a clustered vehicle routing problem

Babiche Aerts Trijntje Cornelissens Kenneth Sörensen

The Joint Order Batching and Picker Routing Problem (JOBPRP) is a very promising approach to minimize the order picking travel distance in a picker-to-parts warehouse environment. We show that this JOBPRP can be modelled as a clustered vehicle routing problem (CluVRP) by replacing vehicles by batches, clusters by orders and customers by pick locations.

To solve this cluster-based model of the JOBPRP, we implement a two-level Variable Neighborhood Search (2level-VNS) meta-heuristic as used earlier for the CluVRP, and study which adaptations are required to perform efficiently in a warehouse environment. Additionally, we test if the Hausdorff distance used for the CluVRP can serve as a valid clustering criterion for order batching. We implement the Hausdorff distance in two different ways in our batching heuristic, and compare the performance with the cumulative minimal aisles visited-criterion, known as a well-performing batching metric in rectangular warehouses with parallel aisles.

Finally, we show that the CluVRP model solved by the 2level-VNS approach performs well compared to state-of-the-art algorithms for the OBP in single-block warehouses. Only a multi-start VNS approach published recently obtains slightly better solutions. Concerning the Hausdorff distance, we must conclude that in most experiments the minimum-aisles criterion retains a better fit in this warehouse context.

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, especially in picker-to-parts systems where the pickers walk through aisles, search and pick items, and bring them to a depot for further consolidation. Additionally, the advent of e-commerce caused a shift from unit-load (pallet) orders to customer orders consisting of various SKUs in small quantities,

making it worthwhile to pick multiple orders in the same pick tour. However, certainly in those e-commerce environments, walking remains by far the most time consuming (De Koster et al., 2007) and minimisation of pickers' travel distances remains a major objective.

The order picker's travel distance is affected by tactical decisions such as layout of the warehouse (*warehouse design*), storage location of the items (*storage assignment*) and assignment of pickers to specific picking areas (*zoning*). But at operational level, the act of clustering customer orders into batches (*batching*) and sequencing the picking of the items in those batches (*routing*), have most impact (Yu and De Koster, 2009). Since large savings on the walking distance can be made by optimising the batching and routing policy simultaneously (Van Gils et al., 2018), we focus in this paper on modelling and solving the combined problem, also known as the Joint Order Batching and Picker Routing Problem (JOBPRP). Although the order batching problem (OBP) and picker routing problem (PRP) have been studied extensively as separate problems, this joint problem is studied to a minor extent.

Just as the PRP bears large similarities with the travelling salesman problem (TSP), so does the JOBPRP resemble the clustered VRP (CluVRP), a variant of the capacitated VRP. The capacitated VRP aims to assign the delivery packages for a group of customers to vehicles of a specific capacity such that the total travel distance of the vehicles is minimised. The CluVRP, introduced by Sevaux et al. (2008), extends this problem by first partitioning customers into clusters based on a predefined criteria (e.g. postal code), and subsequently assigns those clusters to vehicles for which routes are determined. By replacing vehicles by batches, clusters by orders, and customers by pick operations, the JOBPRP can be modelled as the clustered VRP. This structural overlap between vehicle routing and order picking was already highlighted in early literature by Chisman (1975) who introduced the idea of a clustered TSP. Since then, further studies on the cluster-based models are rather found within the VRP literature, apart from Löffler et al. (2018) who recently used the concept to plan the routing in AGV-assisted order picking systems.

In this paper, we model the JOBPRP as a CluVRP, and implement a similar two-level Variable Neighborhood Search (2level-VNS) metaheuristic, as originally developed for the CluVRP by Defryn and Sörensen (2017). This two-level approach is achieved by alternating between a VNS at order-level to compose the batches, and a VNS at picking operation-level to construct the routes. After diversification, the algorithm iterates over the two VNS levels until a stopping criterion is met. In the original CluVRP, Defryn and Sörensen (2017) use the Hausdorff distance as metric for the closeness of customer clusters, while the actual travel distance is not used until the determination of the vehicles' routings. We test whether this Hausdorff distance performs equally well for the JOBPRP, although in contradiction to the CluVRP, in the JOBPRP the same physical pick locations are often visited by different batches.

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 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. We compare the Hausdorff-based batching heuristics with the well-known minimum aisles visited-criterion. In section 5 we describe the 2level-VNS algorithm in detail. The

experimental setup and computational results, as well as a comparison with other 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 e-commerce environments, customer orders consist of one or multiple order lines that each depict a particular item and its 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 (= picking route). To avoid additional sorting operations and associated costs at the depot, items that belong to the same order are not distributed 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 size of a batch is defined by the number of items that fit into the batch size (to the example of Gibson and Sharp (1992), Zhang et al. (2017) and Scholz and Wäscher (2017)). We hereby take into consideration the capacity of the carts that the pickers use to collect the items. The pick tours always start and end at the (single) depot.

Given the objective of minimizing the total travel distance, and knowing the batch size and all orders that have to be picked, two sub-questions remain:

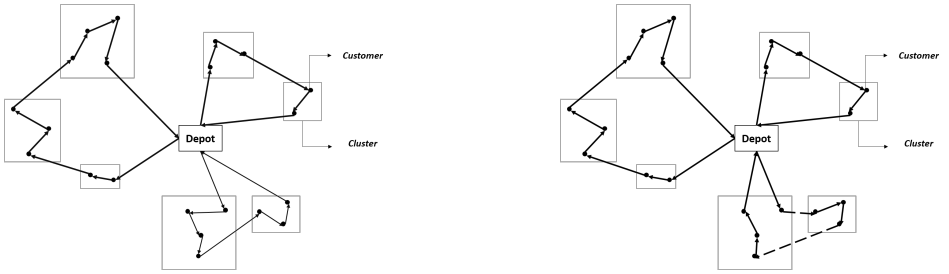
- **Order batching (sub)problem** (OBP): How are orders combined into batches?
- **Picker routing (sub)problem** (PRP): For each batch, in which sequence does the picker visit the pick locations that store the requested items?

When solving the two (sub)problems simultaneously, the combined problem is known as the JOBPRP.

2.2 Comparison with the clustered VRP

In the CluVRP, customers are clustered based on a specific criterion such as geography (e.g. sharing the same postal code) or preference (e.g. preferring the same driver). Next, those clusters are assigned to vehicles of a specific capacity. A cluster can only be assigned to a vehicle if the total demand of all clustered customers fits into the capacity of 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) introduced 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



(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.

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 clustered VRP are illustrated in fig. 1a and fig. 1b.

Many properties of the clustered VRP with soft cluster constraints are recognized in the structure of the JOBPRP and both problems can be defined by the same mathematical model. Indeed, the JOBRPR can be modeled on an undirected graph $G = (V, E)$, where V represents the set of picking operations to be executed. A picking 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 picking operations, and as such as separate nodes. Picking operation V_0 refers to the visit of the depot, which is obligatory at the start and end of each pick route. For each edge $(i, j) \in E$ that connects two nodes (= 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 requesting 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 batches. Each batch has the same 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 order r_0 stands for visiting the depot V_0. The set of pick operations to complete an order r is presented by $C_r = \{V_i \in V \backslash V_0 : r_i = r\}$. Additionally 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 $(i, j) \in V \backslash S \times S$.

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

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

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) is to minimize 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 picking operation, to start their route at the depot. Constraints (4) ensure each location related to a specific pick operation to be entered and left by the same batch. Constraints (5) state that the number of items related to pick operations executed by the same batch, cannot exceed the capacity of that batch. The subtour elimination constraints are described by (6). Constraints (7) refer to the soft cluster constraints, which no longer oblige to fulfil orders one after the other, but allow to complete orders of a batch simultaneously. Lastly, the order integrity rule is represented by constraints (8).

Apart from terminology-based adaptations, the mathematical structure of the JOBPRP and the CluVRP with soft cluster constraints is identical. The difference between both problems is context based and related to the number of visits possible for one location. For the CluVRP, each customer is characterised by a unique location/address that is normally visited only once by a vehicle. For the JOBPRP, there is no restriction on the number of batches that visit a specific warehouse location, since multiple orders can require the same item (stored at the same location). Of course, this number will be minimised for distance purposes, but it is not restricted to one. However, 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 picking operations, where

two picking operations can refer to the retrieval of the same item at the same location but for different orders, retrieved by the same or different batches.

The contextual dissimilarity is also reflected in the data structure of both problems. The CluVRP starts with a list of customers, characterized by a demand and location. During a straightforward precomputation the clusters are created, based on a clustering criterion and the fact that the total demand of a cluster should fit the vehicle's capacity. For the JOBPRP however, the orders are given as input data which already represent the clusters, independent of the storage location of the order's items. Only in a secondary phase information regarding the storage locations of the items is integrated, based on the inventory available in the warehouse. The similarities and differences between CluVRP and JOBPRP are presented in table 1.

CluVRP with soft cluster constraints		JOBPRP
Concepts		
Vehicle	$\leftrightarrow$	Batch
Cluster	$\leftrightarrow$	Order
Customer	$\leftrightarrow$	Pick operation at item level
Input data & precomputation		
List of customers with for each customer: - Customer demand - Customer address - Customer postal code (or preferred driver) Clusters created based on a criterion like postal code (or preferred driver)		
$\Downarrow$		
List of customer clusters with for each cluster: - Cluster demand = sum of customers demand - Customers' address	$\leftrightarrow$	List of orders (each order = a cluster) with: - Order quantity = sum of requested item quantities - Item's number/description
(A)symmetrical routing environment - Available road network	$\leftrightarrow$	Symmetrical warehouse layout - Warehouse layout and 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 JOBPRP.

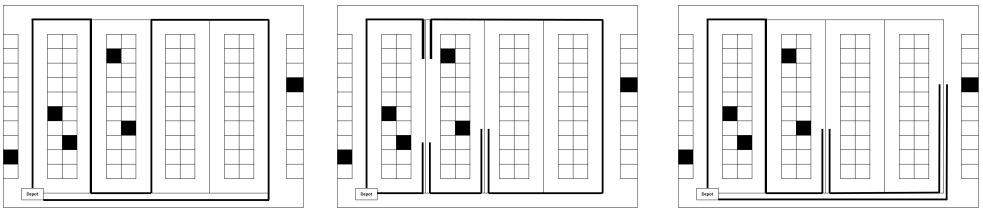
3 Literature review

In section 3.1, we give a literature overview for the PRP and OBP is recapitulated. In next sections, we recapitulate existent literature for the JOBPRP (section 3.2) and CluVRP (section 3.3).

3.1 Warehouse literature on the order batching and picker routing problem

For years the Order Batching Problem (OBP) and Picker Routing Problem (PRP) have been studied as separate problems in warehouse literature. Following and Henn and Wäscher (2012) and Scholz and Wäscher (2017), the OBP is defined as follows: assuming the storage allocation, capacity of the picking device and routing policy to be known, how are orders combined into batches such that the total travel distance is minimized? Sequentially, one solves the PRP resulting for each batch. However, since both OBP and PRP aim to minimise the walking distance for order pickers, these problems are strongly connected (Van Gils et al., 2018). Despite the fact that studies on the joint problem are increasing in the last years (overview given in next section), we observe the majority of experiments have focussed on only one problem. We give an overview of existing research.

In case of a PRP in a rectangular warehouse with parallel aisles, proven to be NP-hard (Won and Olafsson, 2005), the routing for one batch of items is often defined by 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 possible (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 parallel warehouse layout.

Earlier, Ratliff and Rosenthal (1983) developed an exact algorithm to solve the PRP in a single-block warehouse with parallel aisles, which was later extended to a two-block warehouse by Roodbergen and Koster (2001). For larger multi-block layouts, Cambazard and Catusse (2018) are able to solve the PRP in multi-block warehouses up to 8 cross-aisles using a dynamic programming approach, although the complexity of the approach is exponential in the number of cross-aisles. An alternative solution method is available as the PRP shares many similarities with the well-studied Travelling Salesman Problem (TSP). Indeed, the Lin-Kernighan-Helsgaun 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).

Concerning the OBP, Gademann and Velde (2005) have proven that this problem is NP-hard and polynomially solvable when batches consist of only two orders. As a consequence, a large

share of warehouse literature is devoted to the study of heuristics to solve this batching problem for realistic instances. The few exact solution approaches been proposed are limited to small problem instances (up to 32 orders) (Gademann and Velde, 2005). More recently, Öncan (2015) introduced Mixed Integer Linear Programming (MILP) formulations in three variants, where each variant considers another routing policy (s-shape, return and midpoint routing). Optimal solutions were found for small size OBP instances (10 orders per instance).

Ongoing research proves that well-performing batching heuristics often make use of characteristics of the warehouse layout and storage policies. An example is the minimum aisles visited-criterion, used to combine orders with the aim to minimize the number of aisles to be visited by the batch. In fact, the 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 if relevant for this paper.

Seed algorithms create batches one by one, by using specific batching criteria. The seed order of a batch is the first order selected according to a predefined criterion. Next, the same or another criterion is used to assign the following order to the batch (De Koster et al., 2007). These assignments are repeated until the batch is full or the remaining capacity is insufficient to include the smallest unassigned order of the instance. The process continues with the next batch until all orders are assigned. 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 make use of a metric that approximates the closeness between orders instead of taking into account the actual distance between pick locations. Among them the minimum aisles visited-criterion, explained earlier.

Seed algorithms can be implemented in two ways. In *single mode*, each order remains an individual order once added to a batch. Illustrated on the minimal aisles visited-criterion, this means that the number of aisles to complete one order is added to the total, despite a potential overlap of aisles. In *cumulative mode*, the seed order is renewed every time an order is added to the batch. Both orders are merged and the value of the batching criterion is recalculated for the batch as a whole, rather than accumulated for each order separately. The frequent recalculations of the seed order make the cumulative mode more complex and time consuming, but in general the cumulative information has a positive influence on the outcome. Ho and Tseng (2006), Ho et al. (2008) and Van Gils et al. (2018) implemented the minimal aisles visited-criterion in the cumulative mode which they prove to work well in comparison to other studied batching criteria. Despite these arguments, De Koster et al. (1999) showed that the cumulative mode hardly outperformed the single mode for some other 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, as illustrated by Rosenwein (1996). For example, in case of the minimum aisles visited-criterion, the merge is considered beneficial if the combination of two orders results in less aisles to be visited in total.

Metaheuristics have been proposed to solve the OBP in warehouse literature since 2005. Many of those heuristics start with an initial solution that is often created using the savings algorithm discussed above (Henn and Wäscher, 2012). Given this initial solution, the metaheuristic aims to improve the solution by means of neighborhoud 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 those, the Multi-Start VNS approach by Menéndez et al. (2017), was able to outperform previous methods and is considered a state of the art solution method for the OBP.

Most OBP metaheuristics have in common that they explore the neighborhood of the incumbent solution through the execution of moves which are generally accepted when the total travel distance is improved. This means that for each batch re-composition, the total travel distance must be recalculated to decide whether or not the incumbent solution is outperformed. To solve the PRP for each newly composed batch, dedicated routing heuristics are used, for which computational effort is limited. However, 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 that use dedicated routings, perform experiments for a parallel single block warehouse (fig. 2), rather than multiple-block layouts (fig. 3).

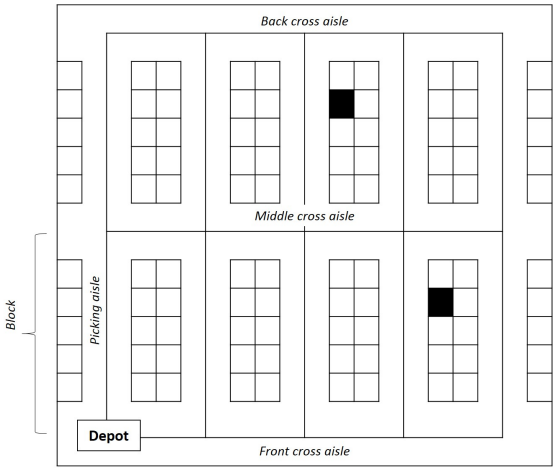


Figure 3: Illustration of a parallel, two-block warehouse.

3.2 Literature on the JOBPRP

Even if much effort is devoted to the composition of good order batches, if these batches are collected by less optimal routings, the shortest total travel distance will not be obtained. There-

fore, it seems straightforward to solve the OBP and PRP in an integrated way, leading us to the joint order batching and picker routing problem (JOBPRP). From our own research, we learned there is a thin line beteen the JOBPRP and OBP. For the sake of clarity, we define that the crucial difference lies in the fact that in the JOBPRP no decision regarding the OBP or PRP is considered to be given. Therefore, metaheuristics that assume a certain routing policy are not found to be applicable for the JOBPRP. We hereby follow Henn and Wäscher (2012), Scholz and Wäscher (2017) and Van Gils et al. (2018).

So far, a handful of metaheuristics have been proposed that solve the joint problem. Among them we find Won and Olafsson (2005), who were among the first to address the joint problem. Kulak et al. (2012) use a tabu search algorithm to solve the order batching problem and two TSP heuristics to solve the resulting routing problems. Batches are created using a route similarity-regret value index (RS-RV) which defines the overlap in travel distances if two orders would be merged. Tsai et al. (2008) present a multiple genetic algorithm approach, one to form the batches, one to construct efficient 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.

So far, Valle et al. (2017) are the only ones to develop an exact approach for the JOBPRP. By means of a branch-and-cut algorithm, instances up to 20 instances were solved to optimality. The authors show that with the cuts presented in their work, computational results were significantly improved. Scholz and Wäscher (2017) present an iterated local search approach in the authors propose a routing heuristic derived from the exact solution approach presented by Roodbergen and Koster (2001). The authors conclude that it is worthwhile to integrate an exact PRP solution instead of simple routing heuristics dedicated to warehouse layout. To our knowledge, Briant et al. (2020) are the latest to contribute to the JOBPRP literature. The authors propose a heuristic based on column generation that provides 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 related to the rectangular warehouse shape. Based on this observation, Briant et al. (2020) claim that also the storage allocation decision could be integrated into the solution approach such that multiple order picking operations can be solved all at once.

3.3 Literature on the CluVRP

A simplified version of the CluVRP was early introduced by Chisman (1975) as the Clustered TSP. The author introduced the idea of clustering customers and assigning those clusters to the vehicle instead of individual customers. He also showed that the same idea can be used in a warehousing context and presented a MILP formulation which he tested for the clustered PRP that results for a one batch-problem. This study forces the so called stock numbers (pick locations) of the same cluster to be visited before visiting the stock number of another cluster, currently known as the strong cluster constraint variant of the problem (visualised in fig. 1a). The author pointed out that these strong cluster constraints might lead to unnecessary travels if clusters contain the same stock number. By guaranteeing that the distance between the

same stock numbers is set to 0, his algorithm forces clusters to be processed consecutively by imposing the last visited stock number of cluster i is sequenced by the same stock number in cluster i+1.

Other studies on the clustered TSP in the warehousing context remain limited to Löffler et al. (2018). This study models the routing of pickers assisted by an AGV (Automated Guided Vehicle) by means of a capacitated clustered TSP. The AGV follows the picker and carries the items retrieved by the picker. Once an order is completed, the AGV returns to the depot while the picker resumes his pick tour, assisted by another, empty AGV. Similar to Chisman (1975), the strong cluster constraints-variant of the problem is applicable (illustrated in fig. 1a). Since the AGV is currently assumed to carry only one order at the time, it is not relevant to look into the soft cluster constraints-variant of the problem.

The majority of further studies on the clustered TSP and by extension the CluVRP are worked out for the vehicle routing context, to begin with Barthélemy et al. (2010) who were the first to approach the problem by means of a heuristic. For an overview of literature on the CluVRP until 2017, we refer to Defryn and Sörensen (2017).

The idea to solve the CluVRP with strong cluster constraints by a two-level VNS, was first proposed by Defryn and Sörensen (2015). The authors claim to reduce the complexity of the problem by first solving the assignment of clusters to vehicles, next solving the routing of the vehicles at customer level while respecting the strong cluster constraints, and finally iterating between those two levels. Indeed, starting from an initial cluster assignment to the vehicles, the first VNS aims to find a local optimum (on a first-improvement basis) through swaps or relocations of clusters for which both inter and intra vehicle operators are implemented. Because the calculation of the travel distance for each potential cluster reassignment is a computational intensive process, the authors proposed the clusters' Euclidean center as a measure for the closeness of customer clusters. During the second VNS, in which the routings of the vehicles are determined at customer-level, the same local search principle is followed, but the swaps and relocations are driven by the travel distance rather than the clusters' Euclidean centres.

The idea of a two-level solution approach was continued by Expósito-Izquierdo et al. (2016) who instead of a VNS propose a combination of the record-to-record travel algorithm using the clusters' Euclidean centres to solve the problem at vehicle-level and the LKH algorithm to solve the problem at customer-level, while respecting the strong cluster constraints. Pop et al. (2018) are the latest to our knowledge to solve the strong cluster constraints CluVRP by a two-level approach. The authors propose a genetic algorithm to solve the problem at cluster-level and the TSP concorde solver (an online solver tool, available at `http://www.math.uwaterloo.ca/tsp/concorde.html`) to determine the corresponding routes.

Both Defryn and Sörensen (2015) and Expósito-Izquierdo et al. (2016) initially propose the clusters' Euclidean center as an approximation for the closeness between customer clusters. Later on, Defryn and Sörensen (2017) suggest the Hausdorff distance as closeness criterion between clusters. The Hausdorff distance is often used for object matching in the field op computer vision and object recognition (Sim et al., 1999), to measure how many similarities two non-empty

sets in a metric space show, in respect of their position (Hung and Yang, 2004). The Hausdorff distance can be used for both Euclidean (e.g. VRP) and Manhattan distances (e.g. PRP).

All solution methods discussed so far tackle the CluVRP with strong cluster constraints. However, Defryn and Sörensen (2017) identified practical arguments for relaxing these constraints and introduced the soft cluster constraint-variant of the problem. By using the same two-level VNS solution method with only minor adaptations, the authors obtained shorter travel distances for the same problem instances.

One of the latest additions to the CluVRP literature with soft cluster constraints, is the development of an exact method by Hintsch and Irnich (2020), who worked out a branch-and-cut algorithm able to optimally solve instances up to 400 customers or 50 clusters.

4 Proposed order batching heuristics

Defryn and Sörensen (2017) originally developed the two-level VNS approach with the aim to integrate the cluster idea into the solution approach. The authors proposed the Hausdorff distance to approximate the closeness between clusters and used this criterion to combine clusters into vehicles, for both the strong and soft cluster constraints-variant of the problem. We will adopt this idea to solve the JOBPRP.

In section 4.1 we interpret the Hausdorff distance as implemented by Defryn and Sörensen (2017), but applied for the case of the JOBPRP. In section 4.2, we adapt the way the Hausdorff distance is used in the batching heuristic in order to align better with the soft cluster characteristics of the JOBPRP. In section 4.3 we explain the minimal aisles visited-criterion that will be used as benchmark for the Hausdorff distance.

4.1 Minimal general Hausdorff distance - as used in the CLuVRP

In case of the JOBPRP, the Hausdorff distance is calculated for each pair of orders (r_i, r_j) as follows: for each item of order r_i, the closest item of order r_j is selected based on a pre-calculated distance matrix that holds the distance between each pair of pick locations. The largest of these distances is eventually selected as the Hausdorff distance between r_i and r_j (Hung and Yang, 2004).

We show an example for five orders, visualised in fig. 4. Each cell refers to a pick location, the number in the cell indicates the order(s) that request an item stored at this pick location. We illustrate the calculation of the Hausdorff distance between r_1 (black coloured cells) and r_2 (cells with shaded pattern). For all locations to be visited for r_1, the closest pick location of r_2 is presented by the thick full lines. Of these three distances, the longest walking distance (8 meters), indicated by H_{12}, is the Hausdorff distance between r_1 and r_2.

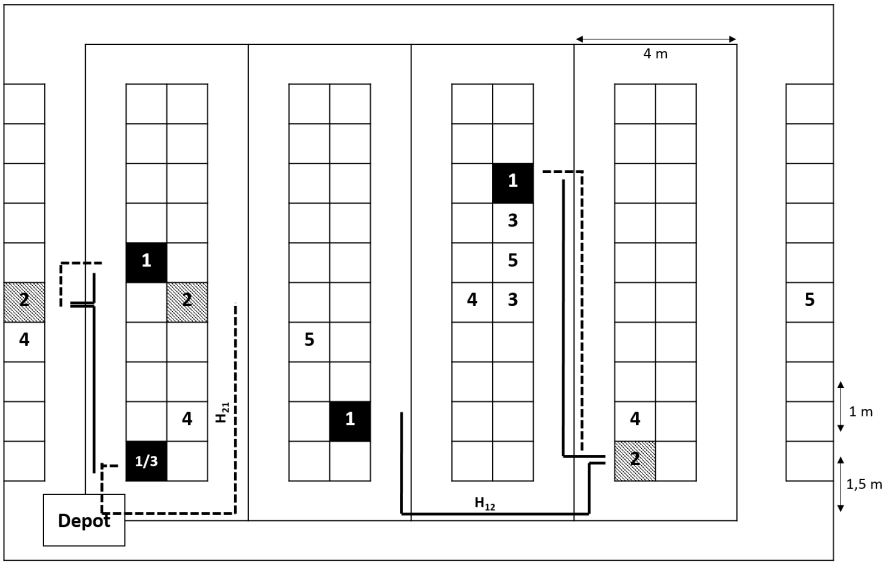


Figure 4: Illustration of the calculation for the general Hausdorff distance between order 1 and order 2. The numbers in the cells represent the items of order 1 to order 5.

Although in a warehouse the distance to walk from location i to j is equal to the distance from location j to i, this is not necessarily true for the Hausdorff distance between two orders. We illustrate this by calculating the Hausdorff distance between r_2 and r_1. The dashed lines in fig. 4 show for each location to be visited for r_2 the closest location of an item requested by r_1. Of these distances, the travel distance indicated by the mark H_{21} is the longest (11 m) and differs from the 8 meter distance for H_{12}. The *general Hausdorff distance* is defined as the maximum of both values, meaning 11 meters for the Hausdorff distance between r_1 and r_2, irrespective of the direction between the orders (Hung and Yang, 2004). Any further use of the Hausdorff measure in the current paper follows the general definition. The general Hausdorff distance between each pair of orders is saved in the symmetrical intercluster distance matrix (see table 2). The last column of table 2 indicates the number of (non-unique) items requested by each order.

	r_0	r_1	r_2	r_3	r_4	r_5	**Nb requested items**
r_0	/	20,5	13,5	19,5	14,5	21,5	
r_1	20,5	/	11	12	8	15	4
r_2	13,5	11	/	11	11	14	3
r_3	19,5	12	11	/	15	15	3
r_4	14,5	8	11	15	/	14	4
r_5	21,5	15	14	15	14	/	3

Table 2: General Hausdorff intercluster distance matrix for the five order-example illustrated in fig. 4.

For the composition of batches using this Hausdorff distance, Defryn and Sörensen (2017) de-

scribe the following procedure. Prior to the actual assignment, orders are sorted in decreasing order based on size (number of requested items), without further distinction between orders requesting the same amount of items. Given the example shown in table 2, 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 and reflects the minimal number necessary to have all orders assigned to a batch (discussed in more detail in section 5.1). For the five order-example, the number of batches is two, both with a capacity of 12 items. Both batches are considered when deciding to which batch an order will be assigned. The first order on the list, r_4, is assigned to the batch of which the last added order is closest to r_4 (i.e. smallest general Hausdorff distance), and with sufficient (remaining) capacity. For empty batches, the depot is considered as the last added order. In the following round, r_1 is added to Batch 1 since r_1 is considered closer to r_4 than to the depot (case for Batch 2). The procedure is repeated until all orders are assigned, leading to the batch composition presented in table 3. The heuristic described above will be denoted as *HausOrig* in the remaining of this paper.

	Order assignment	**Hausdorff distance of the 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 Hausdorff distance		**103 m**
Batch capacity = 12 items		

Table 3: Order assignment for the example in fig. 4 in case of the HausOrig batching heuristic

Because the HausOrig heuristic assigns orders primarily on a size-based criterion, the full potential of the Hausdorff distance as a batching criterion might not be fully exploited. Also, the HausOrig implementation considers only the last order added to each batch when selecting the batch for the next order. Since for the JOBPRP the sequence of orders added to the batch should no longer matter, some opportunities are currently not exploited. Given these remarks, we propose a variant of the HausOrig in the following section.

4.2 Minimal general Hausdorff distance - adapted to JOBPRP

Given the remarks in the previous section, we propose an alternative constructive heuristic that aligns better with the soft cluster constraints-variant of the JOBPRP, from now denoted as *HausAdap*. We continue to use the general Hausdorff distance.

A first modification is to fill batches one by one instead of considering all available batches at once. Orders are added to the current batch as long as the smallest unassigned order fits the remaining capacity, aiming to use all available space in each batch (adopted from Gibson and Sharp (1992), Ho et al. (2008)). Secondly, we drop the sorting operation and use the Hausdorff distance as primary criterion to assign orders to batches, rather than size. However, in case of a tie, preference goes to the largest order.

The third adjustment is to extend the pool of information to decide which order will be assigned next to the current batch. Since the JOBPRP is a soft cluster constraints-variant of the CluVRP, advantage of this property can be taken by considering the Hausdorff distance to all orders already assigned to the batch, instead of considering only the last added order. The depot however is not considered. We argument that since for every batch the inclusion of the depot is obliged anyway, it is more relevant to take into account only fellow orders. The depot is only considered when selecting the seed order.

Working out the HausAdap approach for the example of fig. 4, leads to the batch assignment presented in table 4. r_2 is assigned first because the Hausdorff distance to the depot is the smallest. Given r_2, there are three eligible orders to select next for Batch 1, all having the same Hausdorff distance to r_2. r_1 will be selected since it requests the most items (i.e. the largest order). Once r_2 and r_1 are assigned to Batch 1, we define for both orders the closest unassigned order, and choose the one with the overall smallest Hausdorff. r_4, the order closest to order 1, will be selected and is the last order added to Batch 1 as its remaining capacity is now smaller than the smallest unassigned order. To return to the depot, and finally define the total Hausdorff distance of Batch 1, different possibilities were tested: adding the minimal Hausdorff distance between any order in the batch and the depot, or the maximal Hausdorff distance. Experiments were conducted and showed a clear preference for the latter option. The procedure is repeated for the next batch and this until all orders are unassigned.

	Order assignment	**Hausdorff distance of the 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 Hausdorff distance		**109 m**
Batch capacity = 12 items		

Table 4: Order assignment for the example illustrated in fig. 4 in case of the HausAdap batching heuristic.

For the example visualised in fig. 4, HausOrig leads to a smaller total Hausdorff distance than the HausAdap. This is not always the case, and also irrelevant, it is simply a consequence of the different use of the Hausdorff distance.

We point out that both HausOrig and HausAdapt are performed in a single mode, meaning each order remains an individual order. In the cumulative mode (explained in section 3.2), it is unclear how to calculate and correctly interpret the Hausdorff distance of the final batch. Once all orders are assigned, all orders in the batch are treated as one cumulative order. To define the final Hausdorff distance, the Hausdorff distance between this cumulative order and the depot needs to be included which would be the travel distance between the farthest location visited and the depot. All other visited locations and geographical similarities between orders would not be considered, making the Hausdorff distance deviate from its original definition. Given this argument and the former time-related calculation difficulties, we implement the Hausdorff distance criterion in single mode.

4.3 Minimal aisles visited

The batching heuristic based on the minimum aisles visited-criterion, from now on denoted as *Aisles*, is implemented as a seed-algorithm (batch by batch) in cumulative mode. The customer order for which the least number of aisles have to be visited, is selected as seed. Additional orders are added to the batch such that a minimal number of additional aisles are visited. This batching criterion only takes into account the number of aisles; the distance between aisles is not considered.

The evolution of the intercluster distances in terms of minimum additional aisles to be visited, is presented in table 5 for the example in fig. 4. Stage 1 shows for each order over how many aisles the order is spread. After stage 1, orders collected in a batch are merged to one cumulative order. The intercluster 'distances' for the unassigned orders, on the other hand, are recalculated to represent the number of *additional* aisles to visit.

	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
Stage 4	r_0	/	/	3	/	/	3

Table 5: The intercluster distance in terms of the additional aisles to be visited, worked out for the example of fig. 4, for three consecutive stages of the Aisles batching heuristic.

At the first stage no orders are assigned yet except the depot order r_0. Based on the minimum additional aisles required for each order separately, order r_3 is selected as seed. In stage 2, the additional aisles to be visited on top of those for order r_3 are calculated for each unassigned order. In case of a tie, preference goes to the largest order, which is currently order r_1. In stage 3, orders r_3 and r_1 are now treated as one, and the same procedure is repeated. As Batch 1 has insufficient remaining capacity, we move to Batch 2 for the selection of a new seed (Stage 4). The final batch assignment according to this Aisles batching heuristic 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. 4 in case of the Aisles batching heuristic.

5 Metaheuristic approach for the order batching and picker routing problem

We solve the order batching problem with the 2level-VNS approach as proposed by Defryn and Sörensen (2017). VNS has proven to work efficiently in solving vehicle routing problems (Hansen and Mladenović, 2014) and promising results have been reported for the OBP by Albareda-Sambola et al. (2009) and Menéndez et al. (2017). However, in the latter studies, the VNS was limited to a single level in which batches are constructed and improved, driven by the pickers' travel distance. This distance is obtained assuming a predefined routing policy, as explained in section 3.1.

We apply a VNS at two levels, where the pickers' travel instance is not considered until the second level. 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:

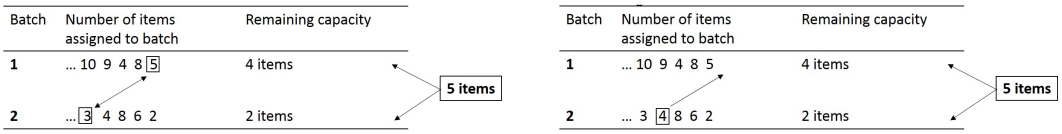
- Warehouse layout (e.g. parallel aisles), number and width of aisles, number and width of cross-aisles, width and depth of pick locations, length of the rack
- Capacity of the batch: equal for all batches, expressed by number of items
- List of orders, with for each order the list of items and required quantity
- 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).

During precomputation, the shortest travel distance between pick locations (including the depot) is calculated and stored in a distance matrix. When the HausOrig or HausAdap heuristic is used, the general Hausdorff distances between orders is determined, using the distance stored in the distance matrix. When the Aisles heuristic is implemented, we compute the number of aisles required to visit for each order separately (stage 1 defined in table 6).

Next, 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) showed that the number of batches used and the total travel distance are strongly related, and also Menéndez et al. (2017) confirm that less batches results in a shorter picking distance. In an attempt to reduce the number of batches from the start, we determine the number of batches K as followed:

$$K = \left\lceil \frac{\text{Total number of items}}{\text{Capacity}} \right\rceil \tag{1}$$

K is clearly a minimum value, but does not guarantee that the order integrity rule is obeyed. We describe in section 5.3 how K is updated if necessary .



(a) Example of an inter batch swap

(b) Example of an inter batch relocation

Figure 5: Illustrations of an inter batch swap and relocation during reassignment, in order to free capacity for an order of five items.

5.2 Step 2: Construction phase

In the 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 is found if all orders are assigned to a batch.

5.3 Step 3: Redistribution phase

After the construction phase, it might occur that not all orders fit into the predefined number of batches. The redistribution function is called to free capacity for unassigned orders by swapping two orders of different batches (i.e., *inter batch swap*, visualised in fig. 5a) or to reallocate an order to another batch (i.e., *inter batch relocation*, visualised in fig. 5b). Both operators are listed in table 7.

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 and add it last in the sequence of another batch Complexity: O(n^2)

Table 7: Moves at order level performed during redistribution.

During this reassignment, orders are reshuffled in order to gain capacity. The orders' size is the primary criterion to do so, the batching criterion does not matter 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 of the other batch. To avoid reassignment moves being reversed directly after execution, a simple tabu list is used which prevents the last move to be undone.

After 10 consecutive reassignments with no improvement (= sufficient clearance of a batch to fit the next order), the number of batches (K) is increased by one. In that case, all orders are reverted back to the status 'unassigned' since a better initial batch assignment might be obtained with the additional batch on hand.

Intra batch operators	
Swap	Swap the position of two orders in one batch's sequence Complexity: O(n^2)
Relocate	Remove an order from the batch's sequence and insert it at another position in the same sequence Complexity: O(n^2)
Two-Opt	Remove two edges from the batch's sequence and replace them by two new edges Complexity: O(n^2)
Inter batch operators	
Swap	Swap two orders belonging to two different batches Complexity: O(n^2)
Relocation	Remove an order from one batch and insert it at a position in the sequence of another batch Complexity: O(n^2)

Table 8: Moves at order-level during intensification phase in case of HausOrig.

5.4 Step 4: Intensification at 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 this batch assignment by a Variable Neighborhood Search (VNS) approach at order-level, based on the batching criterion chosen at the start.

For the original CluVRP approach, using the HausOrig heuristic, the sequence of orders in a batch is of importance, although the characteristics of the JOBPRP suggest otherwise. Therefore, it is relevant to explore inter batch moves (e.g. swapping orders between batches), but also to consider intra batch moves that try to optimize the order sequence within a batch. Defryn and Sörensen (2017) proposed five local search operators that explore inter and intra batch neighbourhoods, as presented in table 8. Numerous combinations of those swaps and relocations are possible, although the computational effort for the evaluation of these operations remains acceptable. An example of one such a move is an inter batch relocation illustrated in fig. 6, where the order positioned second in the sequence of Batch 1 will be relocated 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.

However, to benefit from the soft cluster constraints characteristics of the JOBPRP, we proposed in section 4.2 an alternative constructive heuristic, HausAdap. When implementing this HausAdap approach, we propose the following adaptations in the intensification at order-level. First, we omit the intra batch operators. We argue that the optimization of the orders' sequence has no influence on the routes of the batches composed afterwards and computational effort can be used elsewhere. Secondly, in line with the former argument, we no longer swap or relocate orders to a specific position in an order sequence, but always add them at the end. As a consequence, the amount of moves to be evaluated for the HausAdap is reduced. However, this time benefit only partly compensates for the large complexity of the HausAdap implementation. Since all orders already assigned to a batch must be considered to define the Hausdorff distance of the batch, even the removal of one order can have a large impact.

The operators performed at order-level for the HausAdap heuristic are shown in table 9. The

Relocation		Order assignment	Hausdorff distance
	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 Hausdorff distance *Batch capacity = 12 items*			**103 m**

⇩

Relocation		Order assignment	Hausdorff distance
	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 Hausdorff distance *Batch capacity = 12 items*		Delta = 4 m	**107 m** Do not perform relocation

Figure 6: Illustration of an inter batch relocation in case of HausOrig.

Inter batch operators	
Swap	Swap orders that each belong to a different batch and add last in sequence Complexity: O(n^2)
Relocation	Remove an order from one batch and insert it at the last position in the sequence of another batch Complexity: O(n)

Table 9: Moves at order-level performed during intensification phase for HausAdap.

same operators are applicable for the Aisles heuristic for which the same arguments hold because it is implemented in cumulative mode.

In fig. 7 we illustrate an inter batch relocation of r_1 from Batch 1 to Batch 2. r_1 is removed and added to the end of the sequence instead of inserting it at a specific position, which reduces the possible relocations (before r_3, before r_5, last in the sequence) to only one. However, more computational effort is devoted to the adaptation of the Hausdorff distance. 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, equal to the Hausdorff distance between the depot and r_1. With the removal of r_1, we will have to look for the largest Hausdorff distance between the depot and any of the remaining orders in Batch 1.

During this intensification phase, each move that leads to an improved value of the respective batching metric is accepted. The neighborhoods are checked in a random way by the algorithm. For each neighborhood, all possible moves are evaluated and if no better solution is found, the algorithm randomly picks one of the remaining neighborhoods. If an improvement is found, the algorithm returns to the first neighborhood. The intensification at order-level stops when all neighborhoods were consecutively checked with no improvement.

Relocation		Order assignment	Hausdorff distance
	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 Hausdorff distance *Batch capacity = 12 items*			**109 m**

⇩

Relocation		Order assignment	Hausdorff distance
	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	-11 – 8 – 20,5 + 11 + 14,5	
Delta	Batch 2	+12	
Total Hausdorff distance *Batch capacity = 12 items*		Delta = - 2 m	**107 m** Perform relocation

Figure 7: Illustration of an inter batch relocation when HausAdap is implemented.

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

For each batch constructed in the previous stage, a routing must be determined that visits all pick locations for the orders included in the batch. This routing problem is handled 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 pick operation-level

The intensification process at pick operation-level aims to improve the routes composed in the previous stage. The VNS approach follows the same procedure as described in section 5.4. Both intra and inter batch operators, described in table 10, are implemented, but moves are no longer evaluated by the chosen batching criterion but by total travel distance. Again, when all neighborhoods are consecutively checked with no improvement, the intensification at pick operation-level comes to an end.

Similar to the first VNS, Defryn and Sörensen (2017) make use of both intra and intra batch operators in the VNS at pick operation-level. This means that orders can be swapped or relocated between batches if it leads to an improved total travel distance. In some way it seems strange to include these moves since much attention already has been paid to the swap and relocation of orders at order-level. However, those moves have been evaluated using the batching criterion, while it is possible that further travel distance improvements can be found when the information (position and walking distances) of pick locations is taking into account.

Intra batch operators	
Swap	Swap the position of two pick locations in a single picking tour Complexity: O(n^2)
Relocate	Remove a pick location and insert it at another position in the same picking tour Complexity: O(n^2)
Two-opt	Remove the edges between two pick locations and replace them by two new edges Complexity: O(n^2)
Inter batch operators	
Swap	Swap orders that each belong to a different batch for which one removes all pick locations of the sequence that belong to these orders Complexity: O(n^2)
Relocation	Remove an order's pick locations from the batch's sequence and insert them at a position in the sequence of another batch Complexity: O(n^2)

Table 10: Description of moves performed during intensification phase at pick operation-level

At this point the algorithm has produced an *initial solution.* The total travel distance of the solution, used to evaluate the performance of the algorithm, cumulates the travel distance of all batches.

5.7 Step 7: Diversification phase and iterative loop

After the initial solution has been obtained, the algorithm performs a further exploration of the solution space through diversification. Part of the solution is destroyed and subsequently repaired to comply with feasibility rules. Because of the strong order integrity rule, we perform the perturbation at order-level to ensure orders remain complete. Depending on a predefined parameter, currently fixed at 10%, a number of orders are removed from each batch and reassigned to batches randomly while meeting the capacity constraint. Redistribution (described in section 5.3) is applied 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 (i.e., shorter total travel distance), the incumbent solution is updated. We currently use the same stopping criterion as Defryn and Sörensen (2017): the algorithm ends after 1000 consecutive iterations without improvement.

6 Numerical experiments

In this section, we present the experimental results to evaluate the performance of the proposed two-level VNS (2level-VNS). All experiments are conducted on the dataset developed by Henn and Wäscher (2012). The characteristics of this 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 period.

Finally, we analyse the outcome of the 2level-VNS approach in a twofold way:

- We compare the results for the original Hausdorff (HausOrig), adapted Hausdorff (HausAdap) and Aisles approach to conclude which batching heuristic is superior, and whether this depends on features such as the number of orders or batch size (section 6.3).
- We compare the performance of our approach with algorithms considered state-of-the-art when solving the OBP. In line with Menéndez et al. (2017), we compare our results with their Multi Start-VNS method and with the results of Albareda-Sambola et al. (2009) and Henn and Wäscher (2012) (section 6.4).

Our 2level-VNS algorithm is implemented in C++ Visual Studio 17. All experiments are executed on an Intel(R) Core i7-6820HQ CPU, 2.7 GHz laptop. For each batching criterion, experiments are conducted three times and the average total travel distance is reported.

6.1 Instance description

All experiments are conducted on the instance set developed by Henn and Wäscher (2012). This set 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 by Henn and Wäscher (2012) 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.

The warehouse layout used in our experiments, is a single-block warehouse 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 meter. It takes 5 meter to move from one aisle to the next. To travel from the depot (D) to the first pick location, it takes 1,5 meters (illustrated in fig. 8.

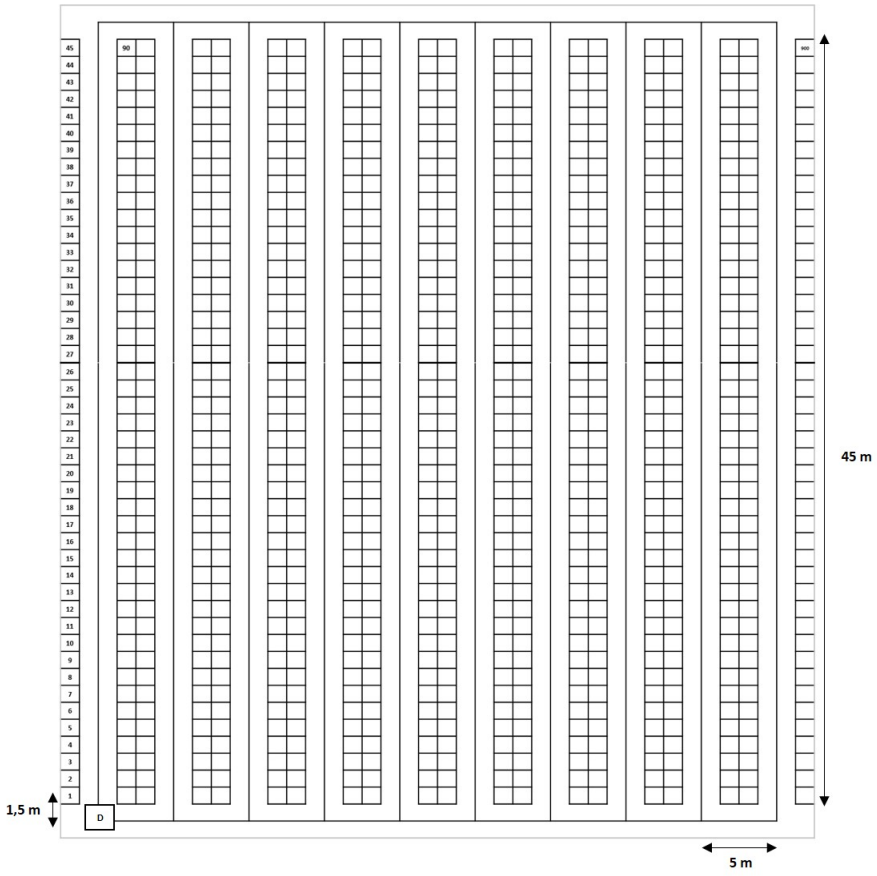


Figure 8: Warehouse layout considered for the instances of Ran_32.

Instances differ in terms of the number of orders (n) and the capacity of the batch (C), expressed in the number of items. For all instances, the number of items per order are uniformly distributed over the values in $\{5,6,\ldots,25\}$. An overview of the values for n and C in the Ran_32 subsets can be found in table 11. Of each (n,C)-combination, two instances are included in Ran_32.

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

Table 11: Overview of (n,C)-values represented in instances of Ran_32 dataset.

Although our 2level-VNS is currently tested for a one-block warehouse, we would like to note that our algorithm can handle instances taking place in multi-block warehouses as well as

layouts other than the often used parallel warehouse layout. All this information is reflected in the distance matrix, it does not change anything to the solution method.

6.2 Preliminary experiments for speeding up the algorithm

We test the 2level-VNS approach for the HausOrig, HausAdap and Aisles on the 32 instances of Ran_32. In table 12 we report the average computation time (in seconds).

		Average CPU (sec) for full algorithm		
	Nb of instances	HausOrig	HausAdap	Aisles
Ran_32	32	**687,72**	**576,51**	**486,07**

Table 12: Average CPU time for the full 2level-VNS algorithm using HausOrig, HausAdap and Aisles respectively, on set Ran_32

Regardless of the batching heuristic, we observe that the computation time is extremely high. Since the JOBPRP is a problem at operational level, the instances should be solved in a realistic time period. Adaptations to the full algorithm seem to be necessary to make it competitive with available (J)OBP(RP) algorithms.

To improve the computation time of our full 2level-VNS algorithm without significant deterioration of the solution quality, we propose three adaptations:

- **Omitting the inter batch moves at routing level**: a large share of computation time is dedicated to the inter batch moves evaluated during the VNS at pick operation-level. Due to the order integrity constraint, all pick operations of an order have to be deleted from one route and put in the best possible position in the route of another batch. All possible sequences are to be evaluated in each iteration, requiring a considerable amount of CPU. Since similar moves have been evaluated during the VNS at order-level, although another acceptance criterion was used, we test if inter batch operators at pick operation-level can be omitted.
- **Omitting the diversification stage**: in this stage, the batch assignments are decomposed and rebuild in a random manner, at least 1000 times. For each new batch composition, intensification at batch- and routing-level are performed in an attempt to find a better solution. As much effort has been devoted to the creation of a good batching criterion and routing heuristic, we will test whether the diversification stage can be left out and stop once an initial solution is found.
- **Adapt the stopping criterion**: instead of omitting diversification, the stopping criterion could be adapted. The current criterion of '1000 consecutive iterations without improvement' takes a lot of time while we question the added value once a certain amount of iterations have been done. We propose to change to a time-based stopping criterion, set to 60 seconds CPU-time for the entire algorithm. Time starts running once the construction stage is initiated (section 5.2).

	Avg. gap total travel distance (%)			Avg. gap CPU-time (%)		
	HausOrig	HausAdapt	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 reflect the average % gap in respect of the solution and computation time realised by full 2level implemenation.

All adaptations were separately tested. In table 13 we show the average solution deviations relative to the full implementation, together with the time improvement. All adaptations, independent of the batching heuristic, reduce the CPU-time significantly, but for the time-based stopping criterion the solution quality deteriorates only little. Hence, we conclude to continue our experiments with the full algorithm, but with a time-based stopping criterion.

To conclude about the duration of the time-based stopping criterion, we performed additional tests. In fig. 9 and fig. 10 the evolution in time of the solution's objective is shown for a random chosen instance, solved with HausOrig and HausAdap respectively. The following observation is independent of the chosen batching heuristic. 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 %) compared to the initial solution, found before diversification. We notice that for both examples, solution improvements are significant during the first minute. Thereafter the solution improves only little.

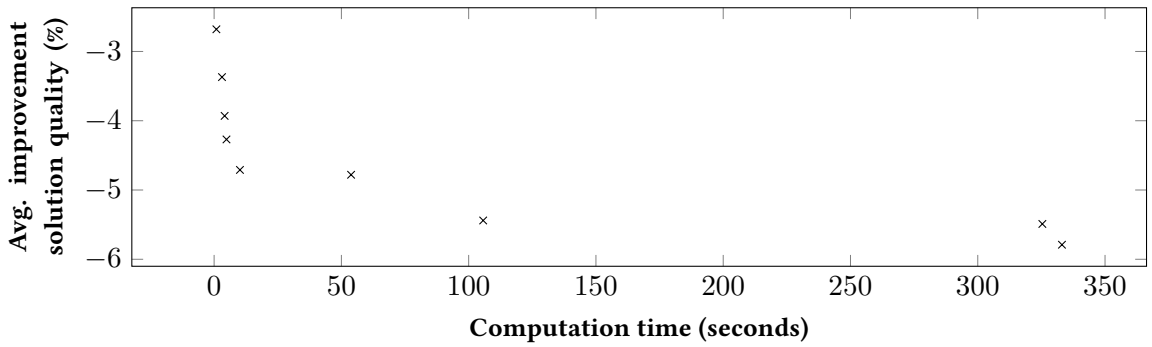


Figure 9: Evolution in time of the solution's objective for instance *40s-60-75-0* and the full implementation with HausOrig. Each mark refers to a better solution, with on the y-axis the % improvement with respect to the initial solution (before diversification).

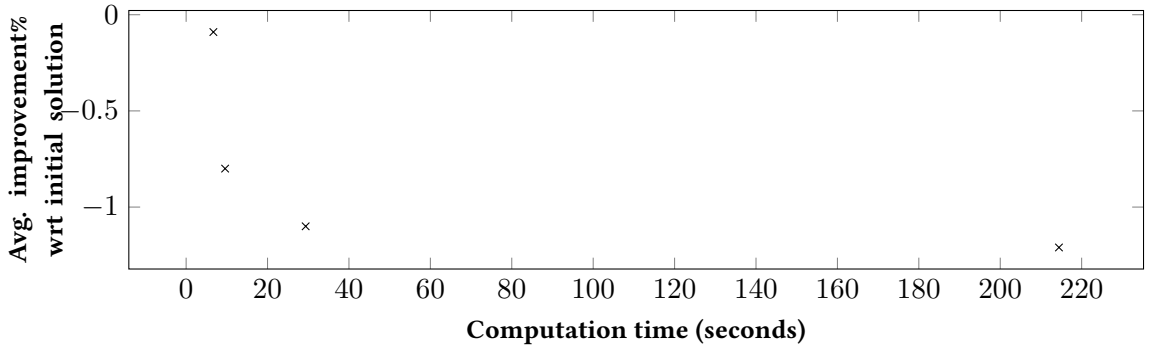


Figure 10: Evolution in time of the solution's objective for instance *40s-60-75-0* and the full implementation with HausAdap. Each mark refers to a better solution, with on the y-axis the % improvement with respect to the initial solution (before diversification).

6.3 Two-level VNS - Comparison of batching heuristics

6.3.1 Validation on limited set, Ran_32

In this section, we analyse the results for the different batching heuristics HausOrig, HausAdap and Aisles for the instances of set Ran_32. In table 14 we compare these batching heuristics pairwise and count for how many instances each of them was able to find the *best solution*, that is the best of both results. This number also includes instances for which both batching heuristics obtained the same solution. For instances for which the criterion performs worse, we calculate the average deviation in respect to the best found solution. In the last column, we

report the average computation time (limited to 60 seconds), which seems to be fully needed for these experiments. Detailed results are provided in Appendix A.

Of both Hausdorff-based batching rules, HausAdap clearly finds the best solution for more instances. Moreover, for the remaining instances the solution is on average only 0,14% worse in contrast to 0,46% for HausOrig. These findings indicate that despite the large similarities between the CluVRP and JOBPRP, our adaptations to the original Hausdorff heuristic are helpful to produce JOBPRP-fit solutions.

Despite these adaptations, we observe that the minimal aisles visited-criterion, included as benchmark, still performs better than the adapted Hausdorff heuristic. The Aisles heuristic was able to find the best solution for 62,5% of the instances. For the remaining instances, the Aisles implementation deviates on average 0,38% from the best found solution, while HausAdap deviates slightly more, 0,54%. Before stating a clear preference, we validate our conclusions by repeating the same pairwise comparisons on a larger dataset. The results are provided in the next section.

		# best solution			**Avg. solution gap (%)**			**CPU (sec)**		
	Nb. Inst.	Haus Orig	Haus Adap	Aisles	Haus Orig	Haus Adap	Aisles	Haus Orig	Haus Adap	Aisles
Algorithm implementation with time-based stopping criterion (60 seconds)										
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 HausOrig, HausAdap and Aisles batching heuristics for the two-level VNS algorithm with time-based stopping criterion of 60 sec.

6.3.2 Validation on larger set of instances, Ran_1280

To validate prior observations we extend our analysis to a larger dataset. In Ran_32, each of the 16 possible (n,C)-combinations (table 110 was represented by two instances. Now, we extract 80 instances for each possible (n,C)-combination from the original set. As such,we obtain a set of 1280 instances, referred to as Ran_1280.

We repeat the previous experiment for dataset Ran_1280, and display the results in table 15. Based on these sets, which statistically include more variation than the small subset, we confirm that the HausAdap outperforms the HausOrig heuristic. Also between HausAdap and Aisles, we find for these larger datasets more obvious preference for the Aisles heuristic, which is able to find a better (or the same) solution for a double amount of instances than HausAdap.

		# best solutions			Avg. gap (%)			CPU (sec)		
	Nb.	Haus Orig	Haus Adap	Aisles	Haus Orig	Haus Adap	Aisles	Haus Orig	Haus Adap	Aisles
Algorithm implementation with time-based stopping criterion (60 seconds)										
Ran_32								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 HausOrig, HausAdap and Aisles batching heuristics for the two-level VNS algorithm with time-based stopping criterion for datasets Ran_640.

Notwithstanding the results in favour of the Aisles heuristic, we do remark that a small subset of instances (32.5%) seems to prefer the HausAdap heuristic. By decomposing the results by number of orders and batch size, we find that especially the batch size influences these results. This decomposition is graphically presented in table 16. Each cell, summarizes the result for one (n,C)-combination. The bars represent the percentage of instances for which HausAdap and Aisles implementation are able to find the best solution.

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 this section by stating that the results we found for the Ran_32 subset align with the results found for the extended dataset Ran_1280. We were able to confirm that the HausOrig heuristic is outperformed by the HausAdap and Aisles heuristic. Regarding the comparison of the HausAdap and Aisles heuristic, we found evidence in the Ran_32 subset to speak of a preference for the Aisles heuristic, which was validated by the larger set Ran_1280, with in particular a larger absolute number of instances for which it performed better than the HausAdap heuristic. However, the dominance of the Aisle heuristic is only visible for 69% of the instances. We therefore continue further experiments with both the HausAdap and Aisles criteria.

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

Our last set of experiments is devoted to the comparison of our 2level-VNS algorithm with the following state of the art algorithms for the OBP:

- Variable Neighborhood Descent (VND) approach byAlbareda-Sambola et al. (2009)
- Attribute-Based Hill Climbing approach with s-shape routing (AHBC + SS) by Henn and Wäscher (2012)

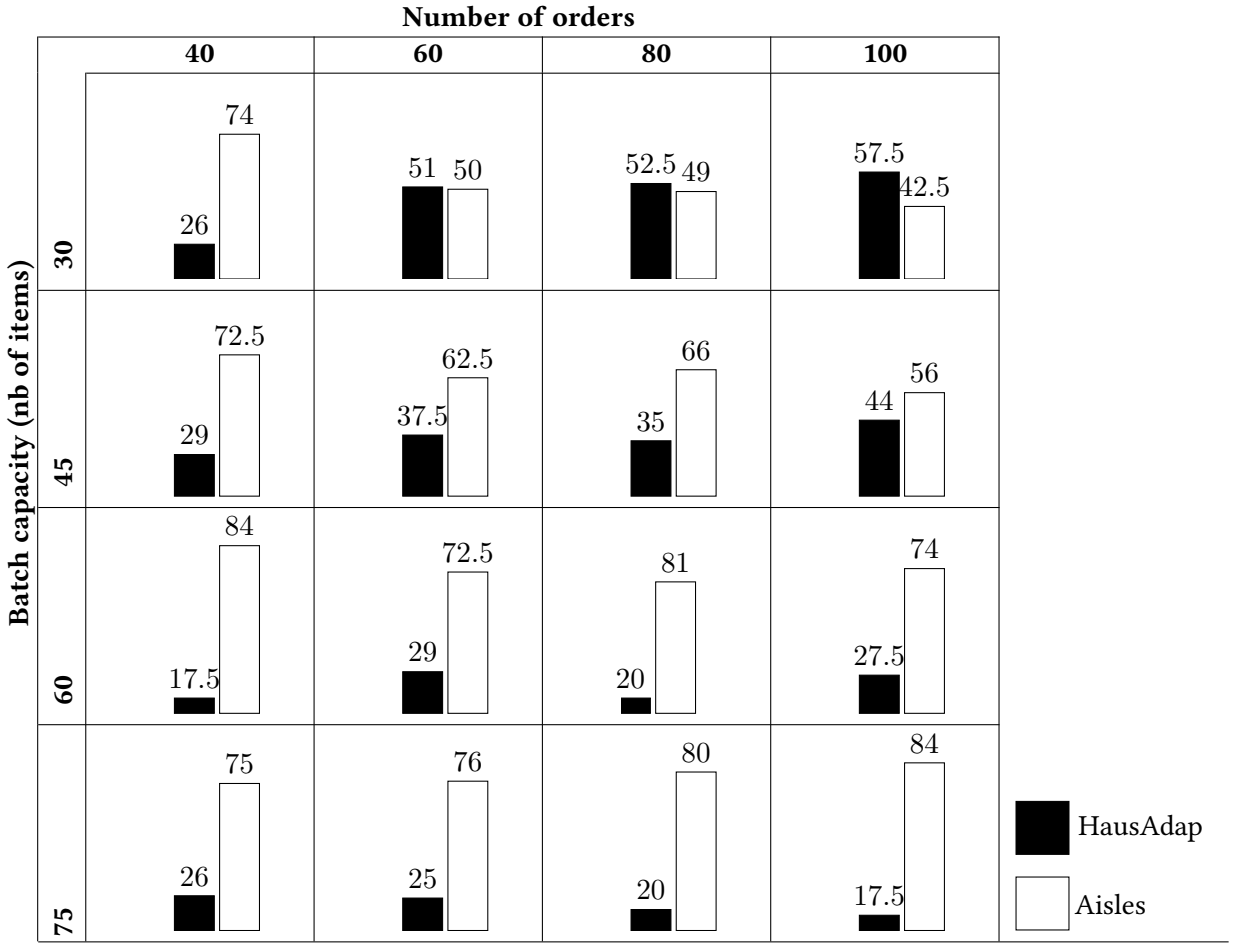


Table 16: Pairwise comparison HausAdap and Aisles heuristic, broken down by number of orders and capacity size. Each cell represents 80 instances.

- 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)

Our comparisons are based on the results made publicly by Menéndez et al. (2017) (available at `http://grafo.etsii.urjc.es/optsicom/obp/`).

For the 32 instances of subset Ran_32, we report the results of the comparison between the 2level-VNS and the four algorithms mentioned above. This comparison is once executed when implementing the HausAdap version in the 2level-VNS, once executed with the Aisles heuristic implemented. In table 17, we first report how many times each algorithm is able to find the best solution out of the results obtained by the five different algorithms. Next, we conduct a pairwise comparison and report how many times the 2level-VNS is able to find the best result as well as the average deviation with respect to the best solution (%). Next we report the average improvement provided by the 2level VNS (%) for each algorithm, in cases where the 2level-VNS performed better. Last but not least, we report the average computation time (in seconds).

In table 17 we show the results when the 2level-VNS is terminated after 60 seconds. Overall, the 2level-VNS performs quite 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-VNS is better for 22 out of the 32 instances, and improves the ABHC+SS solution by 6,29% on average for those 22 cases. Overall results are similar for the HausAdap and Aisles heuristic, although in general, greater improvements were obtained with the Aisles heuristics. Differences however are minor.

When comparing with the MS-VNS, the 2level-VNS performs better for only 8 out of 32 instances. More specifically, from the detailed results in Appendix A, we derive that, independent of the implemented batching heuristic, the 2level-VNS finds regularly better solutions than the MS-VNS method for instances with a smaller batch capacity (30 or 45 items). For larger batch size ($\geq$ 60 items) the MS-VNS consistently performs better.

Finally, in table 18 we provide a similar comparison of the 2level-VNS with the OBP algorithms but used the original stopping criterion, which can be seen from the average CPU reported in the bottom rows. Overall, we find similar results and our conclusions based on table 17 remain valid, confirming that the time-based criterion is a good approach to retain performing solutions that are competitive with state of the art algorithms.

We conclude that for most instances, the 2level-VNS is able to outperform the VND, ABHC+SS and ABHC+LG algorithms, but the results obtained by the MS-VNS remain out of our league, although the solutions deviate to a minor extent. Concerning the batching criteria, we conclude that when the two-level VNS is used, it is best to implement the aisle-based criterion.

		Ran_32		
	# inst.	**Algorithm**	**2level VNS HausAdap**	**2level VNS Aisles**
Overall comparison				
# best solutions	32	**2level VNS**	8	8
		VND	0	0
		ABHC+SS	7	7
		ABHC+LG	0	0
		MS-VNS	17	17
Pairwise comparison				
# best solutions	32	VND	32	32
		ABHC+SS	22	22
		ABHC+LG	32	32
		MS-VNS	8	8
Avg. gap (%) 2level-VNS		VND	-	-
in respect of best found solution		ABHC+SS	2,88%	2,60%
		ABHC+LG	-	-
		MS-VNS	1,81%	1,51%
Avg. improvement (%)		VND	-6,02%	-6,20%
when 2level-VNS better		ABHC+SS	-6,14%	-6,29%
		ABHC+LG	-10,37%	-10,53%
		MS-VNS	-1,51%	-1,38%
Avg. CPU (sec)		2level VNS	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 2level-VNS with stopping criterion 60 seconds, for alternatively HausAdap and Aisles. Results shown for Ran_32.

		Ran_32		
	# inst.	**Algorithm**	**2level VNS HausAdap**	**2level VNS Aisles**
Overall comparison				
# best solution	32	**2level VNS**	9	11
		VND	0	0
		ABHC+SS	0	0
		ABHC+LG	0	0
		MS-VNS	16	14
Pairwise comparison				
# best solution		VND	32	32
		ABHC+SS	22	23
		ABHC+LG	32	32
		MS-VNS	10	11
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. 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. CPU (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 state of the art algorithms for JOBPRP with two-level VNS without time-based stopping criterion, with the HausAdap or Aisles heuristic. Results are shown for the Ran_32 set.

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 to solve 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 warehouse context. In contrary to existing algorithms for the OBP and JOBPRP, this approach does not primarily assign orders to batches based on the total travel distance, but makes use of the Hausdorff distance between orders. However, next to the original implementation of the Hausdorff-based batching heuristic (HausOrig), we developed an adapted version (HausAdap) that benefits from the fact that the JOBPRP can be modelled as a CluVRP with soft cluster constraints. Both HausOrig and HausAdap were compared with the Aisles batching heuristic, based on (a cumulative version of) the minimum aisles visited-criterion, a well known criterion in warehouse context.

We consider the 2level-VNS with adapted Hausdorff batching heuristic as successful, since it was able to find a better solution for twice the number of instances than the original Hausdorff-based implementation. Overall, the aisle-based heuristic is able to find the best solution for 69% of the instance set, in contrary to 32% when adapted Hausdorff is implemented. For the remaining instances, the recorded average deviations are smaller for the aisle-based implementation. We therefore conclude a preference for the latter.

We compared the two-level VNS to state of the art OBP algorithms (VND algorithm (Albareda-Sambola et al., 2009), AHBC + SS and ABHC + LG algorithm (Henn and Wäscher, 2012) and MS-VNS algorithm (Menéndez et al., 2017)). Our two-level VNS outperformed three of the four algorithms, with average improvements between 6% and 10%. However, the MS-VNS algorithm maintains its superiority and performs best, especially for large batch sizes.

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

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

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*.

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.

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.

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

Hansen, P. and Mladenović, N. (2014). Variable neighborhood search. In Burke, E. and Kendall, G., editors, *Search Methodologies*, pages 313–337. Springer, Boston, MA.

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. 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.

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.

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,.

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.

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.

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.

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.

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.

Appendix A. Detailed results

Appendix A shows the detailed results when comparing the 2level-VNS to other JOBPRP algorithms and is complimentary to the data provided in section 6.4. We show the results when alternatively the HausOrig, HausAdap or Aisles heuristic is used, if relevant. We show the total travel distances obtained by the 2level-VNS (in meter), followed by the average deviation (%) in respect of the outcome resulting from the VND, ABHC+SS, ABHC+LG or MS-VNS approach, respectively, found at `http://grafo.etsii.urjc.es/optsicom/obp/`. Table 19 includes the results when the 2level-VNS is implemented using the stopping criterion '1000 iterations without improvement'. In table 20, we provide the results when the 2level-VNS is implemented using a time-based stopping criterion. The time for the entire algorithm is set to 60 seconds.

			Total travel distance (m)	Avg. deviation (%)			
	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%

(continued on next page)

Table 19 - *Continued*

			Total travel distance (m)	**Avg. deviation (%)**			
	n	C	2level-VNS	VND	ABHC+SS	ABHC+LG	MS-VNS
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%
	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%

(continued on next page)

Table 19 - *Continued*

		Total travel distance (m)	**Avg. deviation (%)**			
n	C	2level-VNS	VND	ABHC+SS	ABHC+LG	MS-VNS
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 19: Detailed results for the full 2level-VNS method with stopping criterion '1000 consecutive iterations without improvement' with HausOrig, HausAdap or Aisles heuristic implemented. Results are shown for the Ran_32 instances.

			Total travel distance (m)	Avg. deviation (%)			
	n	C	2level-VNS	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	20972	-8,55%	-8,18%	-5,98%	-0,73%
	100	30	23228	-7,68%	-12,02%	-8,95%	-1,29%
	100	45	14848	-2,81%	1,85%	-3,89%	2,93%
	100	45	14734	-4,88%	-5,42%	-9,35%	2,16%
	100	60	11736	-4,75%	4,80%	-8,31%	2,79%
	100	60	13533	-4,17%	-3,92%	-11,64%	2,50%
	100	75	9694	-4,83%	6,20%	-14,24%	3,18%
	100	75	9884	-5,07%	-1,47%	-13,92%	3,46%
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%

(continued on next page)

Table 20 - *Continued*

		Total travel distance (m)	**Avg. deviation (%)**			
n	C	2level-VNS	VND	ABHC+SS	ABHC+LG	MS-VNS
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%
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 20: Detailed results for the full 2level-VNS method with time-based stopping criterion, time set to 60 seconds, with HausAdap or Aisles heuristic implemented. Results are shown for the Ran_32 instances.