The time-window strategy in the online order batching problem

The time-window strategy in the Online Order Batching Problem

Sergio Gil-Borrás[a], Eduardo G. Pardo[b], Ernesto Jiménez[a], and Kenneth Sörensen[c]

[a]Dept. Sistemas Informáticos, Universidad Politécnica de Madrid, Spain; [b]Dept. Informática y Estadística, Universidad Rey Juan Carlos, Spain; [c]Depart. Engineering Management Antwerpen University, Belgium

ARTICLE HISTORY
Compiled November 20, 2023

ABSTRACT

When an order arrives at a warehouse it is usually assigned to a batch and a decision is made on how long to wait before assigning the batch to a picker and starting the picking tour. If the idle time of the pickers is minimized, the batch is immediately assigned, and the picking starts. Alternatively, if a time window is introduced, other orders may arrive, and more efficient batches may be formed. The method to decide how long to wait (the time-window strategy) is therefore important but, surprisingly, almost completely overlooked in the literature. In this paper, we demonstrate that this lack of attention is unwarranted, and that the time-window method significantly influences the overall warehouse performance. In the context of the online order batching problem (OOBP), we first demonstrate that the effects of different time-window strategies are independent of the methods used to solve the other subproblems of the OOBP (batching and routing). Second, we propose two new time-window strategies, compare them to existing methods, and prove that our methods outperform those in the literature under various scenarios. Finally, we show how time-window methods influence different objective functions of the OOBP when varying numbers of orders and pickers.

KEYWORDS: Online Order Batching Problem, Time Window, Fixed Time Window, Variable Time Window, Order Picking, Warehousing.

1. Introduction

Logistics in a warehouse encompasses a large number of activities. Among them, the collection of orders is one of the most important, due to the high cost associated with this operation compared to the rest of the processes. The operational costs of order picking has been the target for many researchers during the years. Back to the eighties / nineties, it was estimated that the operational costs could represent up to 60% of total costs within a warehouse (Drury 1988; Coyle et al. 1996). Later approaches indicated that labor costs related to the picking process consume about 50-60% of all labor activities in the warehouse (Gademann and Velde 2005; Tompkins et al. 2010). More recent approaches (Shah et al. 2017; Rushton et al. 2022)

CONTACT: Eduardo G. Pardo - Email: eduardo.pardo@urjc.es
This research has been partially supported by grants: RTI2018-094269-B-I00, PGC2018-095322-B-C22, PID2021-125709OA-C22, PID2021-126605NB-I00, and PID2021-122640OB-I00, funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe"; grant P2018/TCS-4566, funded by Comunidad de Madrid and European Regional Development Fund; grant "Programa 466A", funded by Programa Propio de I+D+i de la Universidad Politécnica de Madrid; and grant M2988 funded by Proyectos Impulso de la Universidad Rey Juan Carlos 2022

extended the discussion about the cost of collecting orders in a warehouse. Although the introduction of technology in the warehouses has partially reduced the costs associated with picking (Elsayed and Unal 1989; Kong et al. 2023), it is still identified as the most important operation activity in achieving an efficient warehouse management (Shah et al. 2017; Rushton et al. 2022).

This paper focuses on warehouse order picking systems, which follow the batch order picking policy (i.e., orders are grouped in batches before being collected and orders in the same batch are collected in the same picking tour). In this type of system, there are numerous factors that significantly influence the performance of the collection process. In Petersen (1997), the authors identified the following factors: The layout of the warehouse, which consists of the shape, number of blocks, number of aisles, number of picking positions, etc.; the routing policy, which consists of determining the route that pickers follow within the warehouse to collect the orders; the sorting policy, which consists of determining when and how the picked products are separated in orders; the storage strategy, which determines where to store each type of product; the batching method, which determines how the products are grouped in batches prior to being collected.

Among all existing order picking systems that follow a batching policy (Ho and Tseng 2006; Bozer and Kile 2008; Zhang and Gao 2023; Pardo et al. 2023), we are interested in those which consider a dynamic/online arrival of orders to the warehouse (Li et al. 2016; Yousefi Nejad Attari et al. 2021). Notice that the dynamic order batching systems represent the most common scenario nowadays due to the growth of the e-commerce. In this context, the Online Order Batching Problem (OOBP) represents a family of optimization problems focused on performing an efficient picking operation, by grouping the orders arrived online to the system into batches.

This article focuses on another additional factor, named Time Window (TW), not considered by Petersen, which also has a significant influence on the picking time (i.e., the time that pickers need to perform the picking task) and the completion time (i.e., the picking time together with the waiting time) (Gil-Borrás et al. 2020a). The TW, also known as waiting time in this context, is defined as the time that a picker waits before starting a new picking route. Although at first glance it may seem unnatural (even inefficient) to wait, new orders may arrive during the time window, which could potentially improve the distribution of orders in the batches already created. Furthermore, from a managerial perspective, waiting for the arrival of new orders might result in a more efficient batch configuration. Additionally, the extra time that the picker is waiting can be used for other necessary tasks such as sorting, packaging, labeling, cleaning, refilling, performing administrative work, etc. Therefore, this factor is studied as an additional task to batching, routing, assigning, or sorting, that needs to be determined in the context of problems using a batch collection policy. However, it has received far less attention than other decision problems in the order picking context, such as batching and routing, despite the fact that determining an adequate time window has a large influence on the final overall performance of the method, as we demonstrate in this paper. Determining the waiting time is especially relevant when the arrival of orders is dynamic (i.e., online), although it could make sense also in multi-picker contexts, when all orders are known before starting the pickup (i.e., offline) but there are potential deadlock situations, such as several pickers trying to access simultaneously to a picking position, to a corridor, or just to the depot.

The aim of this work is to answer three research questions. First, we try to determine if we should consider a waiting strategy in any online order batching algorithm. Second, we try to identify which are the best waiting strategies currently existing in the literature. Third, we try to determine which waiting strategy should be used depending on the particular scenario considered. This work is also motivated by the lack of previous works related to determining the time window in the context of the OOBP. Even though there exist works in the literature where the time window is taken into consideration, these are limited to a specific algorithm / function to determine it. Moreover, there is no previous paper focusing on the influence of this task on objective functions related to the OOBP, nor on comparing the most common existent time-window strategies for different scenarios.

This work can be considered as an extension of a previous work presented in Gil-Borrás et al. (2020a). In that work, the authors performed a preliminary study which compared the two main families of time-window strategies: Fixed Time Window and Variable Time Window. However, the authors only considered some basic heuristic strategies for the task. In this

article, we focus on studying, in a detailed way, the influence of the time window on the value of the objective function, in the context of OOBP, depending on different factors such as: Batching, routing, objective function, or congestion in the arrival of orders. Also, we compile and compare the most outstanding strategies for determining the time window in each of the compared scenarios.

The main contribution of this paper is that it demonstrates the positive influence that time-window strategies have on the performance of order batching and related methods. No previous research has demonstrated this influence and, furthermore, only a few papers mention the use of a waiting strategy. We have reviewed the most important time window methods available in the literature, and we have empirically compared them. We also demonstrate the independence of the time window with respect to other tasks such as batching or routing, and we identify the most suitable time-window strategy depending on the objective function studied. Finally, we propose two new time window strategies which complement previous ones.

The rest of the article is organized as follows. In Section 2, the OOBP is described. In Section 3, the state of the art related to the time window is presented, and the most relevant methods are described. Next, in Section 4, two new methods are proposed to determine the time window. In Section 5, it is presented the comparative study carried out in this paper. In this section, it is also reviewed the algorithms for the rest of the tasks that need to be handled in the context of OOBP. In Section 6, the experiments of this paper are presented. Finally, in Section 7, some conclusions and future research lines are pointed out.

2. Online Order Batching Problem

The Online Order Batching Problem is a dynamic optimization problem in which orders continually arrive at a warehouse during the processing time. In this way, not all orders to be processed nor their arrival times are known beforehand. In this context, orders are grouped into batches prior to be picked. The batch collection process can be carried out by a single picker or by several pickers. To solve the OOBP, it is necessary to tackle different tasks / subproblems. First, the orders that reach the system are grouped into batches for later collection. This process of grouping orders into batches is known as *batching*. Once the batches have been generated, a decision must be made on whether it is more convenient to start collecting any of the batches or to wait for new orders to arrive. This process is known as *determining the time window*. The time window can be a fixed amount of time or a variable one (i.e., a different waiting time for each picker on each new route). Furthermore, it is necessary to decide which batch, among all generated ones, is going to be collected next. This process is known as *selecting*. Notice that the term *selecting* is used when only the next batch to collect is chosen, as it is customary in dynamic / online environments, since the conditions might change in the future. On the other hand, the term *sequencing* is used for the same task when all the conformed batches are sorted to determine the order in which they will be picked, as it is customary in static / offline environments. Finally, once the batch and the time for starting the route are decided, it is necessary to determine the route that the picker should follow through the warehouse to perform the picking. This process is known as *routing*. Sometimes, there is an additional process consisting in *sorting* the products collected on the same picking route. This is due to the fact that products belonging to different orders are collected together and they might be placed in the same basket during picking. Therefore, a sorting process must be performed afterward to separate the products into different orders. However, we do not study the influence of this task in our work.

Next, we mathematically define the OOBP. To that aim, in Table 1, we present the parameters and variables used in the model. Then, we define the OOBP considering the minimization of the picking time (i.e., the sum of picking time of all pickers) as an objective function as follows:

$$\min \sum_{j=1}^{m} T_{service}(b_j), \tag{1}$$

subject to,

$$\sum_{j=1}^{m} x_{ji} = 1, \quad \forall\, i \in \{1, \ldots, n\}. \tag{2}$$

$$\sum_{k=1}^{l} y_{jk} = 1, \quad \forall\, j \in \{1, \ldots, m\}. \tag{3}$$

$$\sum_{i=1}^{n} w_i * x_{ji} \leq W, \quad \forall\, j \in \{1, \ldots, m\}. \tag{4}$$

$$st_j \geq \min_{k \in \{1,\ldots,l\}} \max_{s \in \{1,\ldots,j-1\}} y_{sk} * (st_s + T_{service}(b_s)), \quad \forall\, j \in \{2, \ldots, m\}. \tag{5}$$

$$st_j \geq st_{j-1}, \quad \forall\, j \in \{2, \ldots, m\}. \tag{6}$$

$$st_j \geq ar_i * x_{ji}, \quad \forall\, i \in \{1, \ldots, n\}, \text{ and } \forall\, j \in \{1, \ldots, m\}. \tag{7}$$

$$st_j \geq \texttt{tw}(), \quad \forall\, j \in \{1, \ldots, m\}. \tag{8}$$

$$st_j \geq 0, \quad \forall j \in \{1, \ldots, m\}. \tag{9}$$

$$st_j \leq ar_n + \sum_{i=1}^{m-1} T_{service}(b_i), \quad \forall j \in \{1, \ldots, m\}. \tag{10}$$

$$x_{ji} \in \{0, 1\}, \quad \forall\, j \in \{1, \ldots, m\} \text{ and } \forall\, i \in \{1, \ldots, n\}. \tag{11}$$

$$y_{jk} \in \{0, 1\}, \quad \forall\, j \in \{1, \ldots, m\} \text{ and } \forall\, k \in \{1, \ldots, l\}, \tag{12}$$

where `tw`() represents the time-window function that is being evaluated. The constraints in (2) guarantee that each order is assigned to a single batch. The constraints in (3) ensure that each batch is assigned to a single picker. The constraints in (4) verify that the maximum capacity of each batch is not exceeded. The constraints in (5) guarantee that the collection of batch b_j begins once a picker is available. Here, k represents the picker, while s represents the batch

Table 1. Parameters, variables and external auxiliary functions for the OOBP.

Parameters		
n	$\rightarrow$	Number of customer orders received in the system
m	$\rightarrow$	Upper bound of the number of batches (a straightforward value is $m = n$).
l	$\rightarrow$	Number of order pickers.
$v_{routing}$	$\rightarrow$	Routing velocity: number of length units that the picker can traverse in the warehouse per unit of time.
$v_{extraction}$	$\rightarrow$	Number of items that the picker can search and pick per time unit.
t_{setup}	$\rightarrow$	Time that the picker needs to empty the picking device and configure a new route before collecting a new order list.
w_i	$\rightarrow$	Number of items of order o_i for $1 \leq i \leq n$.
W	$\rightarrow$	Maximum number of items that can be included in a batch (device capacity).
ar_i	$\rightarrow$	Arrival time of order i for $1 \leq i \leq n$.

Variables		
st_j	$\rightarrow$	Start time of batch j for $1 \leq j \leq m$.
x_{ji}	$\rightarrow$	$\begin{cases} 1, & \text{if order } o_i \text{ is assigned to batch } b_j, \\ & for\ 1 \leq i \leq n,\ and\ 1 \leq j \leq m. \\ 0, & \text{otherwise.} \end{cases}$
y_{jk}	$\rightarrow$	$\begin{cases} 1, & \text{if picker } p_k \text{ is assigned to batch } b_j, \\ & for\ 1 \leq k \leq l,\ and\ 1 \leq j \leq m. \\ 0, & \text{otherwise.} \end{cases}$

External auxiliary functions		
`tw`()	$\rightarrow$	Time-window function.
`dis`()	$\rightarrow$	Distance function (i.e., routing algorithm).

being collected. The constraints in (6) ensure that the collection of batch b_j starts once the collection of batch b_{j-1} has already started. The constraints in (7) verify that the route to collect a batch b_j cannot start before the timestamps (moments in time) when the orders o_i assigned to that batch have reached the system. The constraints in (8) guarantee that the collection of batch j does not start before the time indicated by the time-window function. The constraints in (9) ensure that st_j is positive. The constraints in (10) ensure that st_j has an upper bound and it is minimum. The constraints in (11) and (12) guarantee that the variables x_{ji} and y_{jk}, respectively, are binary.

$$T_{service}(b_j) = \frac{\texttt{dis}(b_j)}{v_{routing}} + \sum_{i=1}^{n} \frac{w_i x_{ji}}{v_{extraction}} + t_{setup}, \quad \forall j \in \{1, \ldots, m\}, \tag{13}$$

where `dis`() represents the distance function (i.e., the routing algorithm). Similarly, the same problem can be studied by minimizing the completion time of the received orders or the maximum turnover time. In both cases, the only difference with respect to the previous model would be the replacement of the objective function (Eq. 1) with Eq. 14 (in the case of the completion time) or Eq. 15 (in the case of the maximum turnover time).

$$\min \max_{j \in \{1,\ldots,m\}} (st_j + T_{service}(b_j)). \tag{14}$$

$$\min \max_{i \in \{1,\ldots,n\}} \sum_{j=1}^{m} (st_j + T_{service}(b_j) - ar_i) * x_{ji}. \tag{15}$$

Notice that the completion time is determined by the moment in which the picker delivers the last batch, while the maximum turnover time objective function is determined by the turnover time of the order that remains longer in the system.

It is worth mentioning that in this model we summarize all constraints necessary to formulate the order batching problem when considering three different objective functions together, without needing to separate the constraints in three different models to group strictly those necessary to optimize each objective function.

3. State of the art

In this section, we review the chronological evolution of the concept of time window in the literature, which has changed during the years. In addition, we highlight the contribution of each reviewed paper to this concept. Next, we classify and detail the most relevant time-window methods proposed in the state of the art.

The time-window concept emerges in the context of the development of batching algorithms as an additional batching strategy, and it has evolved until being studied as an independent part of the picking process. The first study dates from 1983 (Quinn 1983) where the term time-window batching is used as a strategy to reduce picking time. Il-Choe and Sharp (1991) presented several results for its general application in improving the efficiency of a picking system. Tang and Chew (1997) and Chew and Tang (1999) performed various mathematical analyses related to the use of time-window batching as a batch generation method. Specifically, the upper and lower limits are calculated for the problem, as well as the estimation of the variance and the mean of several objective functions such as travel time, service time, and turnover time. Yu and De Koster (2009) expanded previous works by presenting a model based on queueing theory and introducing the calculation of the concept of "Expected waiting time to form a new batch" ($E[Wj]$) applied to this type of problem. The same year, Van Nieuwenhuyse and De Koster (2009) published an article that estimates the average processing time of an order, using Variable Time-Window Batching (VTWB) or Fixed Time-Window Batching (FTWB) as a batching algorithm. In the same study, the impact of two picking policies (pick-and-sort and sort-while-pick) is also compared for the same Time-Window Batching scenario. This article also develops the concept of $E[Wj]$. Later, in 2012, a new method was introduced in Bukchin et al. (2012) for the first time to accurately calculate the departure time of each picker, assuming that all the arrival times of the orders are known. Based on this informa-

tion, an approximate model is proposed to determine the waiting strategy for future arrivals of orders. The previous article uses Markov Decision Processes and they are compared with a couple of naive heuristics. This work studies the minimization of delays in the delivery of orders, as well as the costs associated with the overtime of the pickers. The same year, Henn (2012) presented a metaheuristic algorithm (Iterated Local Search) for batching, together with a heuristic to calculate the time window.

Xu et al. (2014) presented new results using $E[Wj]$ for the VTWB calculation, together with the First Come First Served (FCFS) algorithm for the batching process. In this paper, the authors calculate the optimal size of batches in this context, for the minimization of the average throughput time. Subsequently, Zhang et al. (2016) first and Zhang et al. (2017) later introduced a rule-based hybrid heuristic, which linked the computation of the time window together with seed-type algorithms for batching. The same year, Giannikas et al. (2017) proposed three variants of a so-called interventionist strategy, consisting in new Variable Time-Window policies based on the number of orders that arrive at the system. Duda and Stawowy (2019), used a Variable Neighborhood Search (VNS) to study the Joint Batch Sequencing and Picker Routing Problem with time windows. In this work, batches are formed beforehand and known in advance. Specifically, they divide the time window into fixed periods of 30 minutes and propose four new neighborhoods to solve the problem in a comprehensive way. Later, Gil-Borrás et al. (2020a) performed a simple comparison between Fixed Time-Window and Variable Time-Window methods. Finally, in the same year, Leung et al. (2020) presented the Intelligent B2B order handling system, solving the Integrated Online Pick-to-sort Order Batching using the Fixed and Variable Time-Window Batching strategies.

Next, we describe the most relevant time-window methods present in the state of the art classified as Fixed Time Window or Variable Time Window methods. Furthermore, some of these methods will be selected to be empirically compared in Section 6.

3.1. *Fixed Time-Window methods in the literature*

Fixed Time-Window (FTW) methods are characterized by determining a fixed waiting time that can depend on different factors of the system / instance, such as the available space in the batch, the distribution of the arrival of orders, or the average number of items of the received orders. Specifically, the strategy that determines the time window could set a single fixed time for all instances of the problem. Alternatively, a FTW strategy could also set a different fixed time for each instance, but depending on a parameter of the instance. The most representative FTW methods in the literature are described below:

- **No-wait method (FTW_NW)**: This is the simplest FTW method. It consists of starting the picking of the next batch as soon as there is a picker available and a batch waiting to be picked. This strategy is one of the most used in the literature. In fact, if not stated otherwise, it is taken as the default strategy. Some examples of works that use this strategy are Gil-Borrás et al. (2020a,b, 2021).
- **Methods with the same fixed time for all instances (FTW_CT)**: It consists of establishing, following any criteria (e.g., a specific time after each picking route, a fixed schedule for departures, etc.), a fixed time that a picker must wait, each time the picker is available, regardless of the specific instance in which the picker is working. This strategy was explored in Gil-Borrás et al. (2020a), where different time-window methods are compared. Specifically, the authors in this paper demonstrate how, in general, this type of method returns worse solutions than methods which use a Variable Time-Window strategy.
- **Methods with a calculated fixed time for each instance (FTW_ZH)**: These methods can be considered as a particularization of the FTW_CT. They consist of setting a fixed time based on a calculation per instance. Specifically, the time window t to start collecting the batch b is calculated as follows: $t = (Q/q) \cdot \lambda \cdot \beta$, where Q is the maximum batch capacity, q is a uniform distribution indicating the number of items of an order, and λ defines the distribution that indicates the arrival rate. Therefore, $(Q/q) \cdot \lambda$ estimates the average time to fulfill a batch. Finally, β is a coefficient of the desired average of fulfillment of a batch before starting the picking process. Notice that β is a search parameter that is studied in Section 6.2.1. An example

of the use of this type of methods was described in Zhang et al. (2017) although it is based on the studies of Xu et al. (2014) and Van Nieuwenhuyse and De Koster (2009). In this work, the authors determined the expected arrival time of the next k orders, required to complete a batch, on the basis of the arrival information of previous orders.

We have selected FTW_NW and FTW_ZH methods to be experimentally evaluated in Section 6 since according to the previous comparison (Gil-Borrás et al. 2020a) FTW_CT methods performed worse.

3.2. *Variable Time-Window methods in the literature*

Variable Time-Window (VTW) methods are characterized by having a variable waiting time, which can be different at each picking moment for the same instance depending on different factors. The most representative methods in the literature are described below:

- **Methods based on a minimum number of orders in queue (VTW_QO)**: This strategy consists of starting the collection of orders in a batch when there is a minimum number of pending orders to be collected in the system. This method was recently used in Gil-Borrás et al. (2020a), where the authors explored the performance of this strategy for different number of queued orders (4, 8, and 16).
- **Methods based on a minimum number of batches (VTW_BA)**: This method sets a different time window depending on the completion of a minimum number of batches previously defined, prior to starting picking. The simplest version of this strategy consists of waiting until the first batch is full. Particularly, when a new batch is configured, it is assumed that the previous batch does not have the capacity to accommodate new orders, so at that time the picker can start collecting the first batch, and so on.
- **Reactive methods according to the collection conditions (VTW_HE)**: This method determines the time window on the basis of the specific system conditions such as the arrival times of the orders and the service times to collect them (Henn 2012). Specifically, if there is more than one batch available for being collected a new route is started as soon as a picker becomes available. Otherwise, the method calculates the moment to start a new route using the following formula: $\max(t, (1+\alpha) \cdot r_i + \alpha \cdot st_i - st_j)$, where t is the current instant time, st_j is the service time of the batch j, st_i is the longest service time of any order i in batch j when it is collected in isolation, and r_i is the arrival time of order i. Finally, α is a coefficient that weighs the arrival time of the order i and the service time of the current batch under construction. Notice that α can be considered as a search parameter and its influence is studied in Section 6.2.2.
- **Methods based on data-built models (VTW_MM)**: In addition to the previously introduced heuristic methods, there is a family of methods based on machine learning models. Specifically, in Bukchin et al. (2012) the authors propose a method that uses Markov decision processes to calculate a subsequent model that determines the optimal moment in which a picker should have started the picking of each batch. Based on the model built with previous data, it establishes a decision-making system for future orders. However, the construction of the exact model presents the difficulty of being a computationally costly task.

According to a previous comparison (Gil-Borrás et al. 2020a), VTW_BA and VTW_HE methods perform better than VTW_QO. Therefore, we have selected the former methods to be experimentally evaluated in Section 6. Additionally, VTW_MM is impractical for real-size instances as the ones used in this paper.

4. New time-window methods

In addition to the previous methods identified in the state of the art, in this work, we propose two new VTW methods to determine the time window, with the aim of finding balance between the effect of waiting in two different objective functions: picking time and completion time.

Particularly, waiting might improve the picking time for a particular batch, but could also deteriorate the completion time of the whole system. The proposed strategies try to determine the expectancy of the next order that will arrive at the system to fit the current batch. These methods have been tested in different empirical scenarios and improve the methods in the state of the art in several of them. The experiments are reported in Section 6.

- **Method based on the available capacity (VTW_M1)**: This method considers that an available picker should wait while there is a batch under construction still incomplete, together with a high estimated probability that the next order will fit in the available capacity. Since the method is based on estimations, in the event that the next order that arrives in the system exceeds the available capacity (i.e., generating a new batch), the picker will start collecting the previous batch. On the other hand, while the capacity is not exceeded after a new arrival, the method constructs a probability distribution based on the size of the orders previously arrived at the system. Based on this distribution, it calculates the probability that the next order will have a smaller size than or equal to the available capacity in the current batch. Then, based on a threshold, the method decides whether to start picking or to wait. The probability threshold is a search parameter and should be adjusted experimentally.
- **Method based on the available capacity with rules (VTW_M2)**: This method could be considered an extension of VTW_M1, but adding two new criteria to start picking: 1) the time calculated for the picking task of the orders currently in the batch under construction is shorter than the time needed until the estimated instant of arrival of the next order; 2) the average time required to pick the items assigned to the batch under construction is at least 10% shorter than the average picking time of items previously collected. Notice that the value of 10% has been empirically tested for this rule.

The VTW_M1 and VTW_M2 methods have been included in the experimental evaluation in Section 6.

5. Description of the comparative study

Evaluating time-window algorithms in the context of the OOBP implies to define the strategies that will be used for the rest of the activities of the problem and the objective function to optimize. For OOBP variants with a single picker, it is necessary to define the batching, selecting, and routing strategies. Moreover, if there are multiple pickers, it is necessary to handle an additional task known as *assigning*.

The batching task consists of grouping a set of orders in a batch, which will be collected together in a single picking route. We consider that orders cannot be split into different batches. There is a great variety of algorithms published in the literature for this task. In this work, we compare three different batching strategies previously proposed in the literature. Particularly, we compare a basic heuristic First Come First Served (FCFS) strategy, widely used as a baseline, an Iterated Local Search (ILS) proposed in Henn et al. (2010), and a General Variable Neighborhood Search (GVNS) presented in Gil-Borrás et al. (2020b). The purpose of exploring different batching strategies is to determine if the time-window method selected depends on the batching algorithm used.

The routing task consists of generating a route that enables the collection of the selected set of orders in a batch within the warehouse. This strategy is also used to determine aspects such as picking time or distance traveled. In this work, four routing algorithms have been compared, the three most popular heuristic algorithms: S-shape (Hall 1993; Petersen 1995), Largest-gap (Hall 1993; Petersen 1995), and Combined (De Koster and Van Der Poort 1998; Menéndez et al. 2017b); and the exact method known as Ratliff and Rosenthal (Ratliff and Rosenthal 1983). The purpose of exploring different routing algorithms is to demonstrate that the time-window strategy is independent of the routing algorithm used.

The selecting method used consists of selecting the batch that contains the largest number of items. In the event that there is more than one batch with the maximum number of items, we select the batch with the shortest picking route. Additionally, the assigning method, which decides which batch is collected by which picker, assigns the next batch to be collected to the first available picker.

In this work, three widely used objective functions for different variants of the OBP are compared: The minimization of the picking time; the minimization of the completion time; and the minimization of the maximum turnover time. The picking time is the time that pickers need to perform the picking task of all orders once the batches are conformed. The completion time is similar to the picking time, but also includes the waiting time for the arrival of new orders. Reducing the picking and completion times, frees some time for the pickers to perform other activities. Also, it helps to reduce energy consumption (in the case that the picking uses machinery) or to reduce the tiredness of the workers (in the case that they walk through the warehouse). Finally, the maximum turnover time is the maximum time that an order remains in the system since it arrives until it is served. Reducing the turnover time results in a direct benefit for the customer since it helps to deliver the products faster.

6. Experiments

This section presents different experiments to evaluate the time-window methods previously detailed in the context of the OOBP. Also, we study the relationship between time-window strategies and other strategies, such as routing or batching algorithms, as well as the influence of either the set instances or the optimized objective function on the performance of the methods.

All methods used in the experimentation, including those of the state of the art, were coded in Java 8 and run on an Intel (R) Core (TM) 2 Quad CPU Q6600 2.4 Ghz computer, with 4 GB DDR2 RAM memory and Ubuntu 18.04.1 64 bit LTS operating system.

In Section 6.1, we present the sets of instances used in the comparison. In Section 6.2, we perform a set of preliminary experiments to adjust the values of the different compared time-window algorithms. Finally, in Section 6.3, we present the final experiments. In those experiments, we compare the behavior of routing and batching algorithms when combined with different time-window strategies.

6.1. *Instances*

Two sets of instances widely used in the state of the art of different variants of order batching problems have been selected for this article. These data sets can be downloaded at https://grafo.etsii.urjc.es/optsicom/oobp/. It is important to note that, to the best of our knowledge, there are no previous specific datasets for "time-window algorithms" in the context of OOBP, since there are no previous studies just devoted to this concept in isolation. However, most of the methods included in our comparison, which deal with the concept of time window, are presented in articles using these data sets. The objective of using two sets of instances is to evaluate whether there is any dependence of the time-window strategies on the set of instances used.

Instances in both data sets correspond to rectangular single-block warehouses with two cross aisles and a single depot. The first data set (Dataset #1) is composed of 80 instances corresponding to 4 different warehouses (denoted as W1, W2, W3, and W4). It was originally proposed in Albareda-Sambola et al. (2009), and it has been used in many related works (Gil-Borrás et al. 2018, 2019, 2020b; Menéndez et al. 2015, 2017c,b). The second data set (Dataset #2) is composed of 64 instances corresponding to a single warehouse (denoted as W5). It was originally proposed in Henn (2012), and it has also been used in many related papers (Aerts et al. 2021; Alipour et al. 2020; Gil-Borrás et al. 2020b; Koch and Wäscher 2016; Menéndez et al. 2017a,b; Pérez-Rodríguez et al. 2015). The detailed characteristics of the instances can be found in Appendix E.

6.2. *Preliminary experiments*

This section is devoted to tuning the parameters of the compared time-window algorithms: FTW_ZH (Zhang et al. 2017), VTW_HE (Henn 2012), VTW_M1, and VTW_M2 (proposed in this paper). To make these adjustments, a diverse subset of 24 instances has been selected

from the original sets of instances previously described. It is worth mentioning that, in the case of VTW_BA, three different configurations (2, 3, and 4 generated batches) are used in the final experiments, denoted as: VTW_B1, VTW_B2, and VTW_B3, respectively.

6.2.1. Selection of parameter β in FTW_ZH

The time-window method proposed in Zhang et al. (2017) uses a β parameter in the formula that determines the fixed time interval of the time window. We refer the reader to Section 3.1 for a detailed description of this parameter. In this experiment, different values of β (0.5, 1, and 1.5) and their influence have been compared. Observing the results, the configuration that obtained the highest number of best solutions in either picking time, completion time, or turnover time was $\beta = 1.5$, with a good performance in the other two indicators. Therefore, we used this configuration for the final experiments. Detailed results for this experiments can be found in Table A1 in the Appendix A.

6.2.2. Selection of parameter α in method VTW_HE

The time-window method proposed in Henn (2012) uses a parameter α to determine the time window before starting the picking of the next batch under construction. We refer the reader to Section 3.2 for a detailed description of this parameter. In this experiment, different values of α (25%, 50%, 75%, and 100%) have been compared. The value 0% for α is not evaluated as it would be equivalent to the FTW_NW method that is already included in the final experiments. The best value found for the parameter α is 50% for two out of the three objective functions studied. For this reason, $\alpha = 50\%$ is selected for the configuration of the method VTW_HE in the final experiments. Detailed results for this experiments can be found in Table A2 in the Appendix A.

6.2.3. Study of the capacity threshold in VTW_M1

The VTW_M1 method determines the probability that the next order arriving in the system will fit into the available capacity in the batch under construction. In this experiment, we study different thresholds to start picking if the probability is below that threshold. Specifically, we have evaluated threshold values between 20% and 70% (in steps of 10%). Extreme values have not been evaluated, as a threshold of 0% indicates that the next order would not fit in the batch, therefore there is no point in waiting for it. Similarly, a value of 100% would equate to the batch being empty. In this case, the best value for this parameter in the three proposed indicators, for the considered objective functions, is 60%. Therefore, this value has been selected for the configuration of VTW_M1 in the final experiments. Detailed results for this experiments can be found in Table A3 in the Appendix A.

6.2.4. Study of the capacity threshold in VTW_M2

The VTW_M2 method determines the probability that the next order arriving to the system will fit into the available capacity (as VTW_M1) but including several additional rules. Specifically, we have evaluated the threshold values between 20% and 70% (in steps of 10%). Again, the extreme values have not been evaluated. In this case, the best value for this parameter, for two out of the three objective functions tested, is 40% for all the proposed indicators, except for the number of best solutions in the case of completion time. Therefore, this value has been selected for the configuration of VTW_M2 for the final experiments. Detailed results for this experiments can be found in Table A4 in the Appendix A.

6.3. *Final experiments*

The aim of this work is to evaluate the behavior of the strategies selected previously to determine the time window in different scenarios. To that aim, we have divided our final experimentation in four sections where we vary: The routing strategy, the batching strategy, or the characteristics of the instance (number of pickers and congestion in the arrival of orders). Our

objectives are: 1) study the influence of the time-window algorithm on the performance of the methods depending on the objective function; 2) identify the best strategy for determining the time window in each scenario; and 3) determine if there exists a dependency of the waiting strategy with respect to either the batching or routing strategies, or other factors such as the number of pickers or the congestion in the arrival of orders.

Notice that all experiments use the same selection and assignment strategies (see Section 5). Additionally, in those experiments where the number of pickers in the warehouse is not explicitly indicated, it is considered that there is only one picker. All experiments in this section have been performed on the whole data set of instances introduced in 6.1. Finally, in Section 6.3.5, we have performed a statistical analysis of the results.

6.3.1. Impact of time-window algorithms on the performance of the studied methods for different objective functions, when combined with several routing strategies

In this experiment, we study the picking time and the completion time objective functions when varying the routing and waiting strategies. Particularly, we combined a GVNS batching method, with four different routing algorithms (S-Shape, Largest-Gap, Combined, and the exact method Ratliff and Rosenthal) introduced in Section 5, and with eight time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) introduced in Section 3 and Section 4.

In Figure 1, we represent the different algorithmic variants compared in terms of completion time and picking time. Particularly, we report the average among all instances of the minimum completion time or the minimum picking time. It can be observed that the time-window method substantially influences the results obtained (either in terms of picking time or in terms of completion time) regardless of the routing method. Also, it is observed that the influence of each time-window strategy on the final result (both in picking time and in completion time) remains constant for each routing method. That is, the best time-window method when using S-Shape, Largest-Gap, Combined, and Ratliff and Rosenthal routing method is always VTW_B3 in terms of picking time. It is also observed that the best time-window method to minimize the picking time (VTW_B3) is not the best time-window method to minimize the completion time. In this case, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH, behave similarly one to each other, and they can be considered the best methods among the compared ones for the completion time.

Similarly, in Figure 2, we compare the performance of the previous methods in terms of maximum turnover time, compared to the completion time. In this case, it is confirmed that, whatever the routing method is, the use of one or another time-window method substantially influences the result obtained in terms of maximum turnover time. Again, we report the average values obtained among all instances. In this case, similarly to what happens with the completion time, the methods VTW_M1, VTW_M2, FTW_NW, and FTW_ZH perform alike and they can be considered as the best ones among the compared methods, for minimizing the maximum turnover time.

It is worth mentioning that, in this case, using an exact method (i.e., Ratliff and Rosenthal) for the routing task does not improve the overall performance of the method. This is due to the fact that calculating an exact route for a particular set of batches provided by the batching algorithm takes longer than calculating an approximate route. Therefore, methods with faster routing procedures are able to explore more solutions than methods which use exact routing procedures, in the same amount of time. Regarding the rest of the methods, the Combined method was the one which performed better. However, since we observed that the influence of the time-window strategy did not depend on the routing method, we selected S-Shape to be used in the following experiments, since it is the most widely used routing method in the literature, and the routes generated are the easiest to follow for pickers (Pardo et al. 2023).

The detailed results to elaborate Figure 1 and Figure 2 can be found in Appendix B.

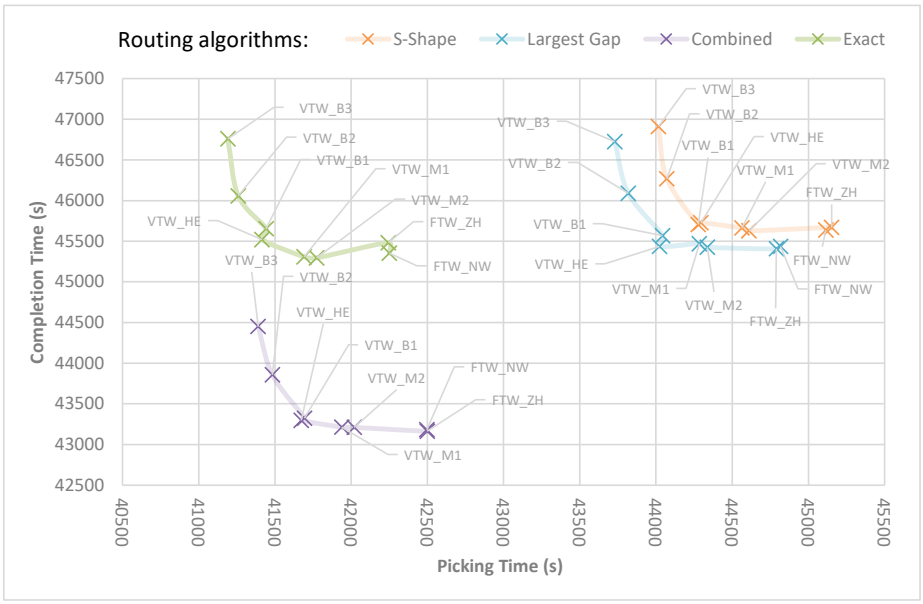

Figure 1. Caption: Behavior of the picking time and completion time when combining the GVNS batching algorithm, with different routing strategies (S-Shape, Largest-Gap, Combined, and Exact) and different time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).
Alt Text: Figure containing an x-y plot comparing the behavior of the picking time and completion time when combining the GVNS batching algorithm, with different routing strategies (S-Shape, Largest-Gap, Combined, and Exact) and different time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).

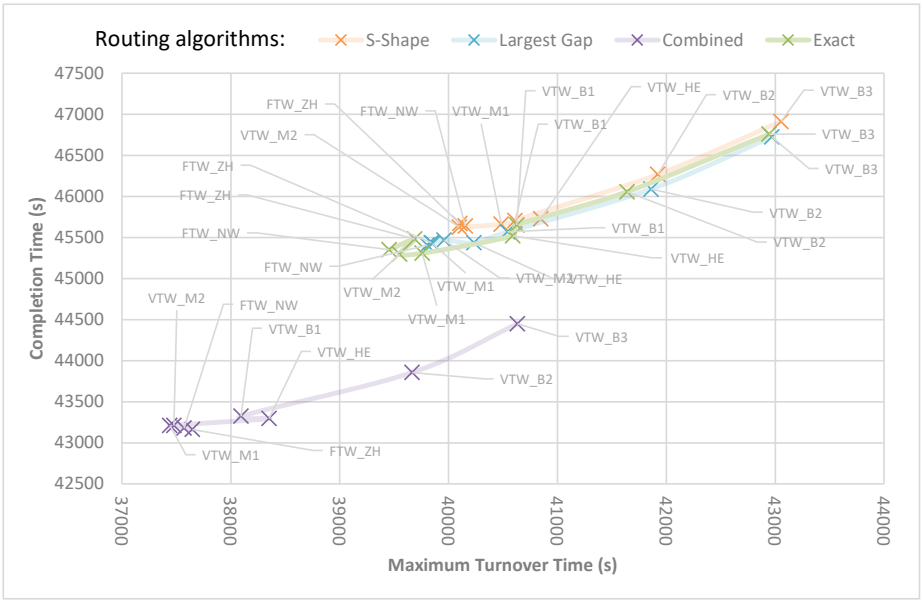

Figure 2. Caption: Behavior of the maximum turnover time and completion time when combining the GVNS batching algorithm, with different routing strategies (S-Shape, Largest-Gap, Combined, and Exact) and different time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).
Alt Text: Figure containing an x-y plot comparing the behavior of the maximum turnover time and completion time when combining the GVNS batching algorithm, with different routing strategies (S-Shape, Largest-Gap, Combined, and Exact) and different time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).

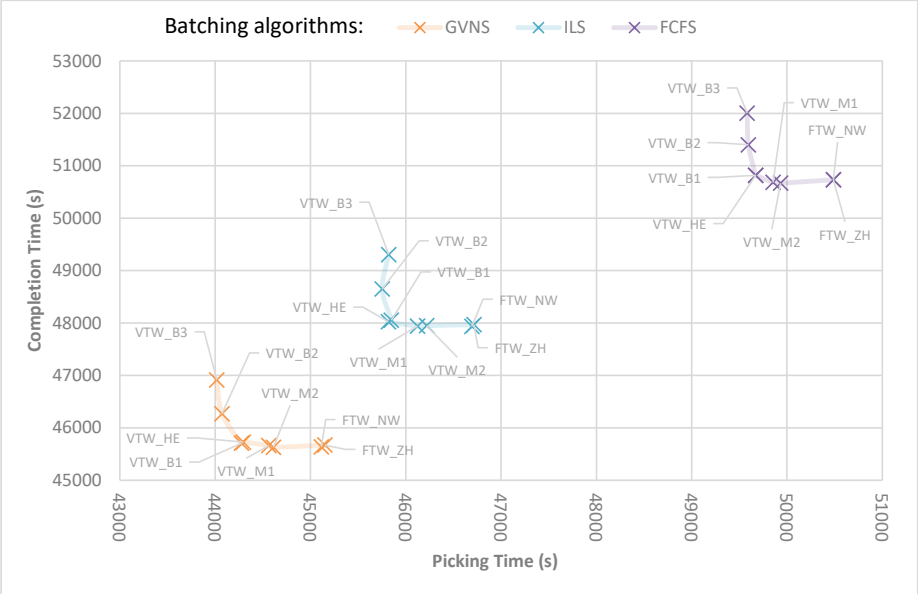

Figure 3. Caption: Behavior of the picking time and completion time when combining different batching algorithms (GVNS, ILS, and FCFS), with several time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the same routing algorithm (S-Shape).
Alt Text: Figure containing an x-y plot comparing the behavior of the picking time and completion time when combining different batching algorithms (GVNS, ILS, and FCFS), with several time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the same routing algorithm (S-Shape).

6.3.2. Impact of time-window algorithms on the performance of the studied methods for different objective functions, when combined with several batching strategies

In this experiment, we study the picking time and the completion time objective functions when varying the batching and waiting strategies. Particularly, we combined three different batching methods (GVNS, ILS, and FCFS), with the S-Shape routing strategy (Notice that, as it was previously mentioned, the S-Shape method introduced in Section 5, is the most used routing strategy in the literature related to OBP), and with eight time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) introduced in Section 3 and Section 4.

In Figure 3, we represent the different algorithmic variants compared in terms of completion time and picking time. As it can be observed, whatever the batching method is, the use of one or another time-window method substantially influences the results obtained (either in terms of picking time or in terms of completion time). Also, it is observed that regardless of the chosen batching method, the influence of each time-window strategy on the final result (both in the picking time and in the completion time) remains constant. That is, the best time-window method for picking time when using GVNS or FCFS is VTW_B3, while VTW_B2 is slightly better than VTW_B3 when combined with ILS. It is also observed that the best time-window method to minimize the picking time (VTW_B3) is not the best time-window method to minimize the completion time. In this case, again, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH are the best methods and behave similarly one to each other for any of the batching methods compared. The detailed results to elaborate Figure 3 and Figure 4 can be found in Appendix C.

Similarly, in Figure 4, we compare the performance of the previous methods in terms of maximum turnover time when compared with the completion time. In this case, it is confirmed that, whatever the batching method is, the use of one or another time-window method substantially influences the result obtained in terms of maximum turnover time. In this case, similar to what happens with the completion time, the methods VTW_M1, VTW_M2, FTW_NW,

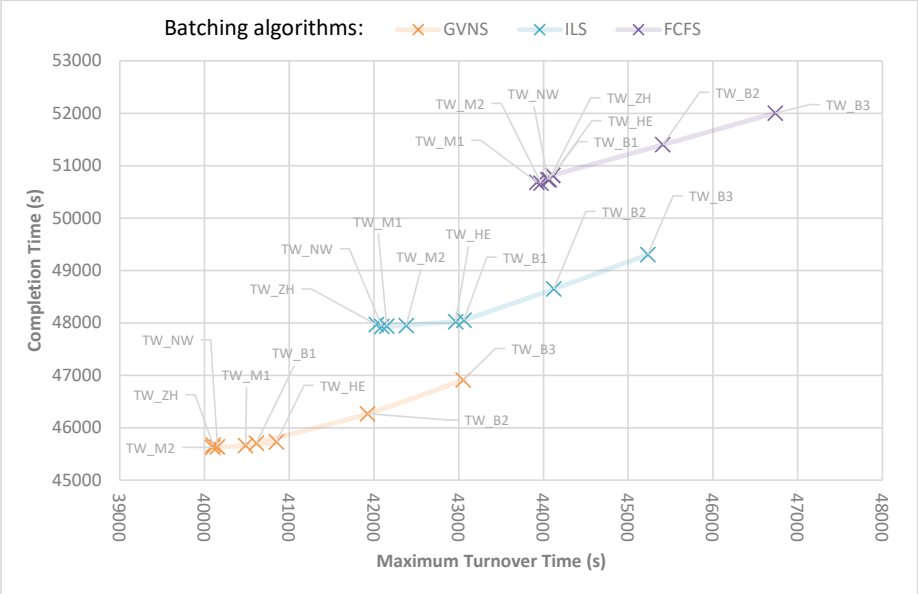

Figure 4. Caption: Behavior of the maximum turnover time and completion time when combining different batching algorithms (GVNS, ILS, and FCFS), with several time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the same routing algorithm (S-Shape).
Alt Text: Figure containing an x-y plot comparing the behavior of the maximum turnover time and completion time when combining different batching algorithms (GVNS, ILS, and FCFS), with several time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the same routing algorithm (S-Shape).

and FTW_ZH perform alike and they can be considered as the best ones among the compared methods, for minimizing the maximum turnover time, for any of the batching strategies compared.

6.3.3. Impact of time-window algorithms on the performance of the studied methods for different objective functions, when varying the number of pickers

In this experiment, we study the picking time and the completion time objective functions, when varying the number of pickers and the waiting strategies. Particularly, we combined the GVNS batching strategy, with the S-Shape routing strategy (all of them introduced in Section 5), and with eight time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) introduced in Section 3 and Section 4 for scenarios with a different number of pickers (1, 2, 3, 4, and 5). The results of this experiment are presented in Figure 5.

As expected, there is an increase in the picking time when increasing the number of pickers, but also a decrease in the completion time. Notice that, more pickers in the system implies that it is more likely that a picker becomes available sooner. Therefore, the queue of orders awaiting to be collected is usually shorter, and orders included in the same batch might generate less efficient routes to collect them. On the other hand, it is observed that the method configured with VTW_B3 is the best in terms of picking time for any scenario (1, 2, 3, 4, and 5 pickers).

As far as the completion time is concerned, the methods FTW_NW, FTW_ZH, VTW_M1, and VTW_M2 present a similar behavior, and can be considered as the best methods for the minimization of this objective function, in any of the scenarios studied (1, 2, 3, 4, and 5 pickers). Notice that the completion time is deeply influenced by the arrival of the last order to be collected. In this scenario, the best strategy to optimize the completion time is to start the last picking tour as soon as possible after the arrival of the last order to the system. This strategy is very similar to a FTW_NW strategy, considering that there is at least one picker available when the last order arrives. Therefore, methods that behaves similarly to the

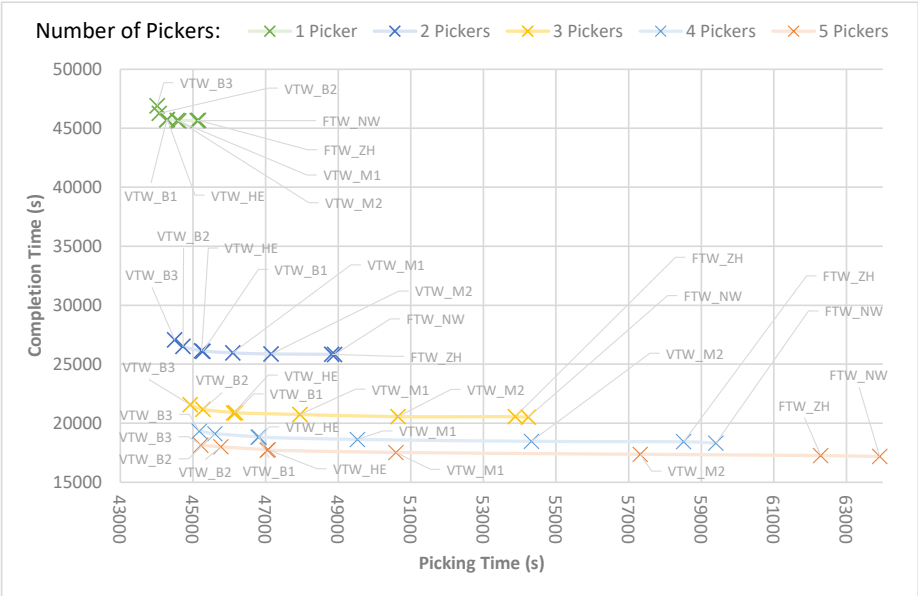

Figure 5. Caption: Behavior of the picking time and completion time when increasing the number of pickers (1, 2, 3, 4, and 5 pickers) for different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH), using a GVNS batching algorithm and a S-Shape routing strategy.
Alt Text: Figure containing an x-y plot comparing the behavior of the picking time and completion time when increasing the number of pickers (1, 2, 3, 4, and 5 pickers) for different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH), using a GVNS batching algorithm and a S-Shape routing strategy.

FTW_NW strategy when the last order arrives are the ones which perform the better.

Finally, it is observed that the differences between the best and the worst methods for the same number of pickers are very small when considering the completion time, regardless of the time-window method. However, in the case of picking time, the differences increase substantially as the number of pickers increases.

Similarly, in Figure 6, the same methods are compared in terms of the maximum turnover time and the completion time. In the case of the maximum turnover time, it is generally observed that it decreases as the number of pickers increases. More specifically, observing the algorithm configured with the same time-window strategy but for a different number of pickers, both the maximum turnover time and completion time decrease as the number of pickers increases. Furthermore, when the number of pickers increases, the influence of the time-window strategy is more relevant in terms of maximum turnover time, since the difference between the worst and the best methods also increases. Derived from the experiments carried out, the methods FTW_NW, FTW_ZH, VTW_M1, and VTW_M2 (which behavior is similar in terms of maximum turnover time) can be considered the best methods for the minimization of the maximum turnover time objective function, in any of the scenarios studied (1, 2, 3, 4, and 5 pickers).

The detailed results to elaborate Figure 5 and Figure 6 can be found in Appendix D.

6.3.4. Impact of time-window algorithms on the performance of the studied methods for different objective functions, when varying the congestion in the arrival of orders

In this last experiment, we compare the picking time, the completion time, and the maximum turnover time for different variants of an algorithm configured with the GVNS batching strategy, the S-Shape routing strategy (both introduced in Section 5), and each of the eight compared time-window strategies (see Section 3 and Section 4), for instances with different number of orders received within the same time horizon (i.e., varying the congestion in the

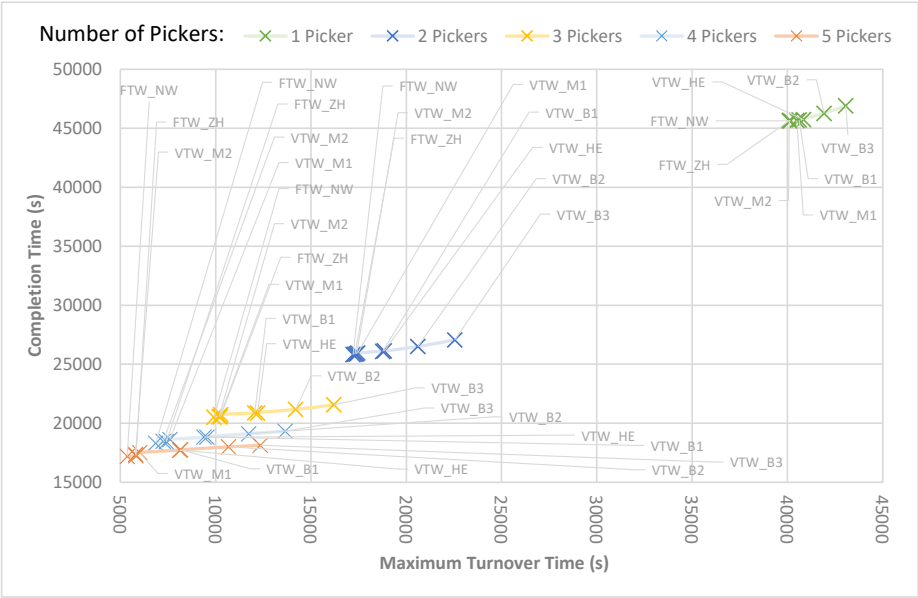

Figure 6. Caption: Behavior of the maximum turnover time with respect to the completion time when the number of pickers is increased (1, 2, 3, 4, and 5 pickers), for several time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH), using a GVNS batching algorithm and a S-Shape routing strategy.
Alt Text: Figure containing an x-y plot comparing the behavior of the maximum turnover time (x) with respect to the completion time (y) when the number of pickers is increased (1, 2, 3, 4, and 5 pickers), for several time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH), using a GVNS batching algorithm and a S-Shape routing strategy.

system). Specifically, the set of instances extracted from Dataset #2 in Henn (2012) has a size which varies from 40 to 100 orders, while the set of instances extracted from Dataset #1 in Albareda-Sambola et al. (2009) has a size which varies from 50 to 250 orders. The results are presented in Tables 2-7 ordered by the objective function (picking time, completion time, and maximum turnover time) and set of instances. Additionally, for each table, the results are organized according to the number of orders.

Table 2. Behavior of the picking time when varying the number of orders (congestion), for several time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the GVNS batching algorithm and the S-Shape routing strategy on the instances extracted from Dataset #2 (Henn 2012).

		Picking time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
40	**Avg (s)**	**15966**	16071	16544	16501	16919	16990	17647	17639
	Dev (%)	**0.22%**	0.94%	4.22%	3.90%	6.67%	7.25%	12.20%	12.16%
	#Best	**10**	5	1	0	0	0	0	0
60	**Avg (s)**	**23066**	23176	23352	23368	23574	23618	24123	24087
	Dev (%)	**0.13%**	0.72%	1.45%	1.63%	2.59%	2.77%	5.57%	5.40%
	#Best	**12**	3	0	0	1	0	0	0
80	**Avg (s)**	**30783**	30843	30956	30931	31134	31061	31372	31423
	Dev (%)	**0.14%**	0.35%	0.72%	0.65%	1.36%	1.19%	2.37%	2.56%
	#Best	**8**	6	0	1	0	0	1	1
100	**Avg (s)**	**36680**	36694	36817	36875	37066	37115	37225	37347
	Dev (%)	**0.28%**	0.31%	0.64%	0.78%	1.32%	1.44%	1.91%	2.18%
	#Best	**7**	**7**	2	0	0	0	0	0

Table 3. Behavior of the picking time when varying the number of orders (congestion), for several time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the GVNS batching algorithm and the S-Shape routing strategy on the instances extracted from Dataset #1 (Albareda-Sambola et al. 2009).

		Picking time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
50	**Avg (s)**	**20042**	20215	20744	20774	21506	21912	23604	23378
	Dev (%)	**0.13%**	1.89%	6.60%	7.34%	14.27%	20.11%	36.03%	35.56%
	#Best	**11**	4	0	0	1	1	0	1
100	**Avg (s)**	**39603**	39757	39917	39919	40172	40256	40655	40633
	Dev (%)	**0.13%**	0.72%	2.13%	2.16%	3.37%	3.80%	6.00%	5.80%
	#Best	**11**	2	0	1	0	1	2	0
150	**Avg (s)**	**58155**	**58115**	58261	58155	58332	58344	58795	58673
	Dev (%)	**0.27%**	0.44%	0.77%	0.65%	1.13%	1.07%	2.19%	1.98%
	#Best	**7**	4	2	0	2	1	0	0
200	**Avg (s)**	**76521**	76615	76588	76693	76823	76747	77155	76872
	Dev (%)	**0.26%**	0.27%	0.49%	0.56%	0.85%	0.76%	1.41%	1.25%
	#Best	3	**6**	2	2	1	0	0	2
250	**Avg (s)**	95337	**95154**	95317	95438	95547	95454	95796	95980
	Dev (%)	0.22%	**0.18%**	0.49%	0.57%	0.61%	0.61%	1.21%	1.21%
	#Best	**7**	4	2	1	0	2	0	0

Table 4. Behavior of the completion time when varying the number of orders (congestion), for several time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the GVNS batching algorithm and the S-Shape routing strategy on the instances extracted from Dataset #2 (Henn 2012).

		Completion time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
40	**Avg (s)**	20527	19578	18511	18527	18336	18328	18281	**18273**
	Dev (%)	14.08%	8.47%	2.12%	2.23%	1.17%	1.14%	0.75%	**0.72%**
	#Best	0	0	0	**4**	3	2	**4**	3
60	**Avg (s)**	25978	25287	24704	24728	24519	24557	24546	**24508**
	Dev (%)	7.60%	4.42%	1.54%	1.72%	0.83%	0.97%	0.91%	**0.75%**
	#Best	0	1	1	1	**6**	3	1	3
80	**Avg (s)**	32982	32390	31832	31811	31742	**31669**	31734	31792
	Dev (%)	5.07%	2.97%	0.90%	0.84%	0.64%	**0.47%**	0.67%	0.87%
	#Best	0	0	1	2	2	4	**5**	2
100	**Avg (s)**	38715	38108	37628	37684	37673	37721	**37624**	37747
	Dev (%)	3.73%	1.88%	0.43%	0.57%	0.53%	0.64%	**0.41%**	0.68%
	#Best	1	0	3	1	2	**4**	2	3

Table 5. Behavior of the completion time when varying the number of orders (congestion), for several time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the GVNS batching algorithm and the S-Shape routing strategy on the instances extracted from Dataset #1 (Albareda-Sambola et al. 2009).

		Completion time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
50	**Avg (s)**	26635	25521	24648	24633	24541	24381	24521	**24288**
	Dev (%)	13.24%	7.63%	2.73%	2.77%	2.06%	1.13%	1.19%	**0.70%**
	#Best	0	0	0	0	2	**7**	3	5
100	**Avg (s)**	42452	41848	41147	41146	41064	41089	40999	**40986**
	Dev (%)	5.93%	3.50%	1.16%	1.21%	0.77%	0.75%	0.65%	**0.51%**
	#Best	0	0	0	1	3	**4**	**4**	**4**
150	**Avg (s)**	59892	59407	59183	**59081**	59094	59113	59230	59114
	Dev (%)	2.67%	1.49%	0.84%	0.72%	0.66%	**0.63%**	0.86%	0.67%
	#Best	0	1	2	2	**5**	2	1	3
200	**Avg (s)**	78173	77922	77474	77581	77590	77490	77670	**77385**
	Dev (%)	1.80%	1.12%	0.45%	0.54%	0.56%	0.45%	0.56%	**0.40%**
	#Best	0	0	2	1	3	2	2	**6**
250	**Avg (s)**	96846	96351	**96224**	96362	96396	96297	96421	96647
	Dev (%)	1.17%	0.55%	0.41%	0.51%	0.41%	**0.40%**	0.66%	0.71%
	#Best	0	**4**	3	3	2	**4**	0	0

Table 6. Behavior of the maximum turnover time when varying the number of orders (congestion), for several time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the GVNS batching algorithm and the S-Shape routing strategy on the instances extracted from Dataset #2 (Henn 2012).

		Maximum turnover time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
40	**Avg (s)**	17100	13604	11433	11847	10025	9628	10458	**9297**
	Dev (%)	154.41%	100.03%	54.76%	60.19%	22.83%	18.30%	23.75%	**7.97%**
	#Best	0	1	2	0	3	3	1	**6**
60	**Avg (s)**	22006	20437	19820	19586	18945	19005	19022	**18690**
	Dev (%)	38.42%	24.04%	19.27%	18.84%	15.23%	13.76%	16.55%	**9.76%**
	#Best	1	0	2	1	2	**5**	2	3
80	**Avg (s)**	27653	27940	26877	27079	27286	27065	**26781**	26917
	Dev (%)	15.17%	15.15%	11.55%	11.85%	12.51%	11.45%	**10.00%**	11.67%
	#Best	**3**	0	1	**3**	1	**3**	**3**	2
100	**Avg (s)**	34425	34161	33085	33300	33989	32864	**32452**	33536
	Dev (%)	15.08%	13.09%	9.29%	10.21%	12.46%	8.17%	**6.46%**	10.48%
	#Best	1	0	1	3	0	**4**	2	5

Table 7. Behavior of the maximum turnover time when varying the number of orders (congestion), for several time-window strategies (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) using the GVNS batching algorithm and the S-Shape routing strategy on the instances extracted from Dataset #1 (Albareda-Sambola et al. 2009).

		Maximum turnover time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
50	**Avg (s)**	23219	21542	17319	16898	15952	15490	**14223**	15420
	Dev (%)	361.54%	311.83%	144.50%	128.32%	33.13%	23.38%	**4.51%**	9.54%
	#Best	0	0	1	2	0	4	**7**	4
100	**Avg (s)**	37576	36891	34410	35450	**34145**	34973	34912	34827
	Dev (%)	48.68%	42.05%	18.58%	23.60%	**9.18%**	10.92%	15.57%	14.37%
	#Best	0	0	3	2	**4**	2	**4**	1
150	**Avg (s)**	56722	55610	55806	54965	55661	**55196**	55506	55520
	Dev (%)	8.80%	5.59%	4.83%	5.17%	4.28%	**3.42%**	4.62%	4.59%
	#Best	0	1	1	2	3	**6**	0	3
200	**Avg (s)**	75048	74190	**73723**	74862	74633	74113	74592	74075
	Dev (%)	6.22%	5.10%	**3.11%**	5.29%	4.78%	4.88%	4.65%	3.98%
	#Best	0	2	**4**	1	2	3	1	3
250	**Avg (s)**	93737	92925	93034	93645	93731	**92555**	92986	93099
	Dev (%)	4.08%	2.52%	2.19%	2.97%	2.65%	**2.08%**	3.03%	2.17%
	#Best	1	1	1	1	3	**4**	3	2

Notice that the congestion in the system increases when, on the same time horizon, the number of orders received is increased. Thus, it is observed that the picking time, the completion time, and the maximum turnover time also increase with the increase of congestion.

For the same number of orders, in all evaluated scenarios, we observe significant differences between the worst and best time-window methods. On the other hand, the differences between the methods are greater with fewer orders in the same time horizon. As far as objective functions are concerned, it can be stated that the differences between the best and worst methods are more accentuated in the case of minimizing the maximum turnover time. This situation can be partially explained by the fact that it looks for the minimization of a maximum value.

Finally, in the case of minimizing the picking time, regardless of the size of the instance, the best time-window method was VTW_B3. Similarly, for completion time, the most competitive methods were: VTW_M2, FTW_ZH, and FTW_NW. Finally, in the case of maximum turnover time, the best methods were: VTW_M1, VTW_M2, FTW_ZH, and FTW_NW.

6.3.5. Statistical analysis

To end this empirical comparison, we performed several statistical tests to corroborate if the differences found among the methods are statistically significant. In this case, we have selected the use of non-parametrical tests, since the samples do not follow a normal distribution. Particularly, we have used Friedman's rank test for comparing the differences among all algorithms together, and the Wilcoxon's test for comparing pairs of algorithms. Therefore, we have compiled all the different executions performed in this final experimentation, and we have performed a Friedman Rank Test. The obtained p_value of 0.000, reported by the statistical software, indicates that there are differences among the methods. In Table 8 we report the rank reported by the test, either separated by objective function, or considering all of them together.

Further than the overall differences, we would like to observe if there are differences between the methods that are closely ranked. To that aim we have performed a pairwise comparison of the methods using the Wilcoxon's Signed Test. The two variants of the proposed methods in this paper VTW_M2 and VTW_M1 do not show statistically significant differences (with a p_value higher than 0.05) between them, however, there are differences between the two proposed methods and any other of the compared ones, including the third one in the ranking VTW_HE. Also, we observed that there are almost no differences between: VTW_HE and VTW_B1, VTW_B1 and VTW_NW, and VTW_NW and VTW_ZH.

Table 8. Friendman's Rank Test.

Picking Time		Completion Time		Turnover Time		Aggregated	
Method	**Rank**	**Method**	**Rank**	**Method**	**Rank**	**Method**	**Rank**
VTW_B3	2,54	**VTW_M1**	3,35	**VTW_M2**	3,72	**VTW_M2**	4,09
VTW_B2	2,79	**VTW_M2**	3,36	**FTW_NW**	3,73	**VTW_M1**	4,12
VTW_HE	3,66	**FTW_NW**	3,49	**VTW_M1**	3,74	**VTW_HE**	4,26
VTW_B1	3,73	**FTW_ZH**	3,53	**FTW_ZH**	3,80	**VTW_B1**	4,29
VTW_M2	5,08	**VTW_HE**	4,16	**VTW_B1**	4,39	**FTW_ZH**	4,38
VTW_M1	5,24	**VTW_B1**	4,3	**VTW_HE**	4,51	**FTW_NW**	4,46
FTW_NW	6,43	**VTW_B2**	6,31	**VTW_B2**	5,68	**VTW_B2**	4,96
FTW_ZH	6,53	**VTW_B3**	7,51	**VTW_B3**	6,43	**VTW_B3**	5,44

7. Conclusions and future work

In this work, we have studied the influence of the time-window strategy on the performance of the algorithms for the OOBP. Specifically, we have reviewed the evolution of the concept of time window in the context of the OOBP during the years. Additionally, we have reviewed the most relevant previous strategies for determining the time window in the literature and selected the most prominent ones to be empirically evaluated. Also, we have proposed two new variable time-window strategies.

Our main conclusion is that it can be stated that there is a relevant influence of the time-window algorithm on the performance of the methods for the OOBP when any of the following

objective functions are studied: Picking time, completion time, or maximum turnover time. Also, we have identified that the time-window algorithm should be determined depending on the objective function tackled, since there is no efficient time-window algorithm for all objective functions.

We have also demonstrated that the influence of the strategy to determine the time window on the objective function does not depend on the batching or routing methods used. It has also been shown that the selection of the best time window for each scenario does not depend on the number of orders processed. However, when considering multiple pickers, the larger the number of pickers, the greater the differences among the compared methods, when minimizing the picking time or the turnover time. On the other hand, increasing the number of pickers results in a slight decrease of the differences among the compared methods, when studying the completion time.

The results obtained in the experimentation carried out during this study indicate that, in online contexts, the minimization of the picking time can be in conflict with the minimization of the completion time. This is supported by the fact that, when minimizing the picking time, we look for batches that can be efficiently collected. To this end, it might be necessary to wait longer for the arrival of new orders, so batches can be better composed. On the contrary, waiting longer for the arrival of new orders implies that the completion time is increased. This circumstance does not occur in offline contexts where the picking time and the completion time are aligned. Therefore, in future works related to the OOBP, the completion time and the picking time could be studied using a multiobjective approach.

The results obtained in this work indicate the importance of properly defining the time-window strategy within the OOBP context. On average, we observed that for a single picker and an arrival time horizon of 4 hours, it is possible to save up to 17 minutes in the picking time, 21 minutes in the completion time, and 49 minutes in the maximum turnover time, simply by choosing an appropriate time-window algorithm. Furthermore, some of the above savings increase as the number of pickers also increases. For example, with 5 pickers, the differences in the picking time, choosing one or the other time-window method, can be up to 311 minutes (> 5 hours).

Among the methods evaluated for determining the time window, when minimizing the picking time, it is recommended to use the VTW_B3 method, while in the case of minimizing the completion time or the maximum turnover time, it is recommended to use one of the following methods: VTW_M1, VTW_M2, FTW_ZH, or FTW_NW. Despite the fact that some methods in the literature perform very well for specific objective functions, the methods proposed in this paper (VTW_M1 and VTW_M2) have a very good behavior for any of the objective functions studied in this paper.

Finally, we would like to highlight the importance of choosing adequately the time-window method depending on the studied objective function, but also on the context conditions (such as the number of pickers or the congestion of the system). On the contrary, the selection of a time-window method does not depend on the batching or routing method chosen.

Given the relevance of the time window in the context of the order batching, presented in this work, it might be interesting to further explore the most suitable scenarios for each time-window method. Also, this should be extended to the study of new time-window proposals and their influence on other objective functions related to the OOBP, not studied in this work, such as minimizing the blocking time or minimizing the cost. Additionally, it would be interesting to study the effect of the dynamic update of the batches, even if the tour to collect them has already started.

Data availability statement

The authors confirm that the data supporting the findings of this study is freely available upon request and in the Appendix of this paper.

References

Aerts, B., Cornelissens, T., Sörensen, K., 2021. The joint order batching and picker routing problem: Modelled and solved as a clustered vehicle routing problem. Computers & Operations Research 129, 105168.

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

Alipour, M., Mehrjedrdi, Y.Z., Mostafaeipour, A., 2020. A rule-based heuristic algorithm for on-line order batching and scheduling in an order picking warehouse with multiple pickers. RAIRO-Operations Research 54, 101–107.

Bozer, Y.A., Kile, J.W., 2008. Order batching in walk-and-pick order picking systems. International Journal of Production Research 46, 1887–1909.

Bukchin, Y., Khmelnitsky, E., Yakuel, P., 2012. Optimizing a dynamic order-picking process. European Journal of Operational Research 219, 335–346.

Chew, E.P., Tang, L.C., 1999. Travel time analysis for general item location assignment in a rectangular warehouse. European Journal of Operational Research 112, 582–597.

Coyle, J.J., Bardi, E.J., Langley, C.J., 1996. The management of business logistics. volume 6. West Publishing Company Minneapolis/St. Paul, Minneapolis-Saint Paul, Minnesota, USA.

De Koster, R.B.M., Van Der Poort, E.S., 1998. Routing order pickers in a warehouse: a comparison between optimal and heuristic solutions. IIE Transactions 30, 469–480.

Drury, J., 1988. Towards more efficient order picking. IMM monograph 1.

Duda, J., Stawowy, A., 2019. A VNS Approach for Batch Sequencing and Route Planning in Manual Picking System with Time Windows. Lecture Notes in Computer Science. International Conference on Variable Neighborhood Search 12010, 167–177.

Elsayed, E.A., Unal, O.I., 1989. Order batching algorithms and travel-time estimation for automated storage/retrieval systems. The International Journal of Production Research 27, 1097–1114.

Gademann, N., Velde, V.S., 2005. Order batching to minimize total travel time in a parallel-aisle warehouse. IIE Transactions 37, 63–75.

Giannikas, V., Lu, W., Robertson, B., Mc. Farlane, D., 2017. An interventionist strategy for warehouse order picking: Evidence from two case studies. International Journal of Production Economics 189, 63–76.

Gil-Borrás, S., Pardo, E.G., Alonso-Ayuso, A., Duarte, A., 2018. New VNS Variants for the Online Order Batching Problem. Lecture Notes in Computer Science 11328, 89–100.

Gil-Borrás, S., Pardo, E.G., Alonso-Ayuso, A., Duarte, A., 2019. Basic VNS for a variant of the Online Order Batching Problem. Lecture Notes in Computer Science 12010, 17–36.

Gil-Borrás, S., Pardo, E.G., Alonso-Ayuso, A., Duarte, A., 2020a. Fixed versus variable time window warehousing strategies in real time. Progress in Artificial Intelligence 9, 315–324.

Gil-Borrás, S., Pardo, E.G., Alonso-Ayuso, A., Duarte, A., 2020b. GRASP with Variable Neighborhood Descent for the Online Order Batching Problem. Journal of Global Optimization 78, 295–325.

Gil-Borrás, S., Pardo, E.G., Alonso-Ayuso, A., Duarte, A., 2021. A heuristic approach for the online order batching problem with multiple pickers. Computers & Industrial Engineering 160, 107517.

Hall, R.W., 1993. Distance approximations for routing manual pickers in a warehouse. IIE Transactions 25, 76–87.

Henn, S., 2012. Algorithms for on-line order batching in an order picking warehouse. Computers and Operations Research 39, 2549–2563.

Henn, S., Koch, S., Doerner, K.F., Strauss, C., Wäscher, G., 2010. Metaheuristics for the order batching problem in manual order picking systems. Business Research 3, 82–105.

Ho, Y.C., 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, 3391–3417.

Il-Choe, K., Sharp, G.P., 1991. Small parts order picking: design and operation. Technical Report. School of Industrial and Systems Engineering. Georgia Institute of Technology. Atlanta, Georgia, EEUU.

Koch, S., Wäscher, G., 2016. A grouping genetic algorithm for the Order Batching Problem in distribution warehouses. Journal of Business Economics 86, 131–153.

Kong, X.T., Zhu, M., Liu, Y., Qin, K., Huang, G.Q., 2023. An advanced order batching approach for automated sequential auctions with forecasting and postponement. International Journal of Production Research 61, 4180–4195.

Leung, K.H., Lee, C.K.M., Choy, K.L., 2020. An integrated online pick-to-sort order batching approach for managing frequent arrivals of b2b e-commerce orders under both fixed and variable time-window batching. Advanced Engineering Informatics 45, 101–125.

Li, J., Huang, R., Dai, J.B., 2016. Joint optimisation of order batching and picker routing in the online retailer's warehouse in China. International Journal of Production Research 55, 447–461.

Menéndez, B., Bustillo, M., Pardo, E.G., Duarte, A., 2017a. General Variable Neighborhood Search for the Order Batching and Sequencing Problem. European Journal of Operational Research 263, 82–93.

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

Menéndez, B., Pardo, E.G., Duarte, A., Alonso-Ayuso, A., Molina, E., 2015. General variable neighborhood search applied to the picking process in a warehouse. Electronic Notes in Discrete Mathematics 47, 77–84.

Menéndez, B., Pardo, E.G., Sánchez-Oro, J., Duarte, A., 2017c. Parallel variable neighborhood search for the min-max order batching problem. International Transactions in Operational Research 24, 635–662.

Pardo, E.G., Gil-Borrás, S., Alonso-Ayuso, A., Duarte, A., 2023. Order batching problems: Taxonomy and literature review. European Journal of Operational Research DOI: `https://doi.org/10.1016/j.ejor.2023.02.019`.

Pérez-Rodríguez, R., Hernández-Aguirre, A., Jöns, S., 2015. A continuous estimation of distribution algorithm for the online order-batching problem. The International Journal of Advanced Manufacturing Technology 79, 569–588.

Petersen, C.G., 1995. Routeing and storage policy interaction in order picking operations. Decision Science Institute Proceedings 31, 1614–1616.

Petersen, C.G., 1997. An evaluation of order picking routeing policies. International Journal of Operations & Production Management 17, 1098–1111.

Quinn, E.B., 1983. Simulation of order processing waves. Material Handling Focus 83.

Ratliff, H.D., Rosenthal, A.S., 1983. Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem. Operations Research 31, 507–521.

Rushton, A., Croucher, P., Baker, P., 2022. The Handbook of Logistics and Distribution Management: Understanding the supply chain. Kogan Page Limited, London, UK.

Shah, B., Khanzode, V., et al., 2017. A comprehensive review of warehouse operational issues. International Journal of Logistics Systems and Management 26, 346–378.

Tang, L.C., Chew, E.P., 1997. Order picking systems: Batching and storage assignment strategies. Computers & Industrial Engineering 33, 817–820.

Tompkins, J.A., White, J.A., Bozer, Y.A., Tanchoco, J.M.A., 2010. Facilities planning. John Wiley & Sons, Hoboken, New Jersey, USA.

Van Nieuwenhuyse, I., De Koster, R.B.M., 2009. Evaluating order throughput time in 2-block warehouses with time window batching. International Journal of Production Economics 121, 654–664.

Xu, X., Liu, T., Li, K., Dong, W., 2014. Evaluating order throughput time with variable time window batching. International Journal of Production Research 52, 2232–2242.

Yousefi Nejad Attari, M., Ebadi Torkayesh, A., Malmir, B., Neyshabouri Jami, E., 2021. Robust possibilistic programming for joint order batching and picker routing problem in ware-

house management. International Journal of Production Research 59, 4434–4452.
Yu, M.M., De Koster, R.B.M., 2009. The impact of order batching and picking area zoning on order picking system performance. European Journal of Operational Research 198, 480–490.
Zhang, J., Wang, X., Chan, F.T.S., Ruan, J., 2017. On-line order batching and sequencing problem with multiple pickers: A hybrid rule-based algorithm. Applied Mathematical Modelling 45, 271–284.
Zhang, J., Wang, X., Huang, K., 2016. Integrated on-line scheduling of order batching and delivery under b2c e-commerce. Computers & Industrial Engineering 94, 280–289.
Zhang, K., Gao, C., 2023. Improved formulations of the joint order batching and picker routing problem. International Journal of Production Research 61, 7386–7409.

Appendix A. Detailed preliminary experiments

In this appendix we compile the detailed results for the preliminary experiments performed to adjust the parameters of the compared methods. In all tables, Avg.(s), stands for the average value of the objective function measured in seconds; Dev.(%), stands for the deviation to the best solution of the experiment; and #Best, stands for the number of best solutions found.

Table A1. Study of different values for the β parameter in FTW_ZH method for picking time, completion time, and turnover time.

		β value		
		0.5	**1.0**	**1.5**
Picking time	**Avg.(s)**	**41291**	41300	41293
	Dev.(%)	**0.17%**	0.31%	0.18%
	#Best	10	9	**13**
Completion time	**Avg.(s)**	41955	41967	**41950**
	Dev.(%)	0.20%	0.36%	**0.19%**
	#Best	7	7	**10**
Maximum turnover time	**Avg.(s)**	36385	36157	**36180**
	Dev.(%)	**6,55%**	7,28%	8,61%
	#Best	7	7	**10**

Table A2. Study of different values for the α parameter in VTW_HE method for picking time, completion time, and turnover time.

		α value			
		25%	**50%**	**75%**	**100%**
Picking time	**Avg.(s)**	40515	**40430**	40483	40518
	Dev.(%)	0.61%	**0.51%**	0.57%	0.73%
	#Best	6	**12**	7	5
Completion time	**Avg.(s)**	42121	**42022**	42085	42111
	Dev.(%)	1.06%	**0.83%**	0.97%	1.02%
	#Best	3	**7**	3	3
Maximum turnover time	**Avg.(s)**	36895	37254	38035	**36887**
	Dev.(%)	**7,46%**	10,06%	14,67%	9,44%
	#Best	6	5	2	**11**

Table A3. Study of different threshold values in VTW_M1 method for the picking time, completion time, and turnover time.

		Threshold (%)					
		20%	**30%**	**40%**	**50%**	**60%**	**70%**
Picking time	**Avg.(s)**	40738	40793	40779	40856	**40721**	40813
	Dev.(%)	0,24%	0,38%	0,36%	0,31%	**0,21%**	0,33%
	#Best	7	4	4	6	**8**	5
Completion time	**Avg.(s)**	41725	41776	41760	41839	**41701**	41799
	Dev.(%)	0,23%	0,37%	0,34%	0,30%	**0,19%**	0,34%
	#Best	4	1	5	**6**	**6**	2
Maximum turnover time	**Avg.(s)**	36792	36541	36314	36851	36469	**35910**
	Dev.(%)	15,51%	14,45%	15,85%	14,61%	13,85%	**7,42%**
	#Best	1	2	4	5	4	**8**

Table A4. Study of different threshold values in VTW_M2 method for the picking time, completion time, and turnover time.

		Threshold (%)					
		20%	**30%**	**40%**	**50%**	**60%**	**70%**
Picking time	**Avg.(s)**	40359	40335	**40285**	40334	40310	40295
	Dev.(%)	0,43%	0,47%	**0,27%**	0,36%	0,34%	0,41%
	#Best	**6**	4	**6**	**6**	5	3
Completion time	**Avg.(s)**	40858	40838	**40784**	40837	40815	40790
	Dev.(%)	0,41%	0,47%	**0,28%**	0,35%	0,35%	0,39%
	#Best	4	**5**	3	**5**	**5**	2
Maximum turnover time	**Avg.(s)**	38408	39093	38720	38589	38641	**38316**
	Dev.(%)	**4,20%**	9,35%	5,52%	6,73%	7,56%	5,15%
	#Best	6	2	**7**	2	4	3

Appendix B. Detailed results of the influence of time-window algorithms on the performance of the studied methods for different objective functions, when combined with different routing strategies

Table B1. Behavior in terms of picking time of a GVNS batching algorithm combined with different routing strategies (S-Shape, Largest-Gap, Combined, and Exact) and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).

		Picking time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
S-Shape	**Avg (s)**	**44017**	44071	44277	44295	44564	44611	45152	45115
	Dev (%)	**0,20%**	0,64%	1,95%	2,03%	3,57%	4,33%	7,65%	7,57%
	#Best	**76**	41	9	5	5	5	3	4
Largest Gap	**Avg (s)**	**43730**	43818	44043	44023	44284	44337	44790	44817
	Dev (%)	**0,15%**	0,76%	2,04%	2,02%	3,54%	4,43%	7,47%	7,46%
	#Best	**92**	26	4	10	3	6	3	2
Combined	**Avg (s)**	**41390**	41483	41694	41673	41939	42020	42499	42497
	Dev (%)	**0,13%**	0,62%	1,91%	1,87%	3,46%	4,35%	7,73%	7,74%
	#Best	**89**	28	6	14	3	2	2	2
Exact	**Avg (s)**	**41191**	41259	41442	41416	41691	41774	42244	42251
	Dev (%)	**0,17%**	0,70%	1,87%	1,82%	3,48%	4,40%	7,93%	7,92%
	#Best	**85**	28	8	12	3	5	1	2

Table B2. Behavior in terms of completion time of a GVNS batching algorithm combined with different routing strategies (S-Shape, Largest-Gap, Combined, and Exact) and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).

		Completion time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
S-Shape	**Avg (s)**	46911	46268	45706	45728	45662	**45627**	45670	45638
	Dev (%)	6,14%	3,56%	1,18%	1,23%	0,85%	0,73%	0,74%	**0,67%**
	#Best	1	6	12	15	28	**32**	22	29
Largest Gap	**Avg (s)**	46724	46090	45568	45436	45469	45425	**45406**	45436
	Dev (%)	6,28%	3,73%	1,46%	1,14%	0,99%	0,87%	**0,73%**	0,75%
	#Best	0	1	8	10	**35**	32	27	33
Combined	**Avg (s)**	44450	43856	43324	43296	43213	43210	**43163**	43181
	Dev (%)	6,22%	3,61%	1,28%	1,21%	0,79%	0,70%	**0,48%**	0,50%
	#Best	1	2	13	11	32	28	25	**37**
Exact	**Avg (s)**	46759	46058	45653	45521	45308	**45296**	45481	45352
	Dev (%)	6,87%	4,11%	1,75%	1,64%	1,17%	1,05%	1,00%	**0,81%**
	#Best	2	5	9	17	**34**	26	24	27

Table B3. Behavior in terms of maximum turnover time of a GVNS batching algorithm combined with different routing strategies (S-Shape, Largest-Gap, Combined, and Exact) and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).

		Maximum turnover time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
S-Shape	**Avg (s)**	43054	41922	40612	40848	40485	**40099**	40104	40154
	Dev (%)	72,49%	57,71%	29,79%	29,60%	13,00%	10,71%	9,90%	**8,28%**
	#Best	6	5	16	15	18	**34**	23	29
Largest Gap	**Avg (s)**	42970	41860	40542	40233	39960	39853	**39804**	39843
	Dev (%)	69,43%	48,89%	30,86%	27,69%	11,44%	11,09%	**9,43%**	9,69%
	#Best	6	6	18	20	**26**	21	24	25
Combined	**Avg (s)**	40633	39665	38094	38351	**37440**	37477	37647	37571
	Dev (%)	75,02%	55,63%	30,91%	29,59%	10,07%	10,41%	9,12%	**8,96%**
	#Best	8	6	16	14	24	23	25	**30**
Exact	**Avg (s)**	42944	41639	40627	40590	39759	39550	39690	**39455**
	Dev (%)	71,02%	51,31%	33,58%	35,01%	11,05%	9,12%	7,39%	**6,72%**
	#Best	3	13	14	17	23	26	**27**	22

Appendix C. Detailed results of the influence of time-window algorithms on the performance of the methods for different objective functions, when combined with different batching strategies

Table C1. Behavior in terms of picking time of different batching algorithms (GVNS, ILS, and FCFS) combined with S-Shape routing strategy and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).

		Picking time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
GVNS	**Avg (s)**	**44017**	44071	44277	44295	44564	44611	45152	45115
	Dev (%)	**0.20%**	0.64%	1.95%	2.03%	3.57%	4.33%	7.65%	7.57%
	#Best	**76**	41	9	5	5	5	3	4
ILS	**Avg (s)**	45820	**45752**	45849	45818	46123	46218	46715	46693
	Dev (%)	**1.15%**	1.22%	2.05%	1.96%	3.38%	4.38%	7.60%	7.58%
	#Best	41	**42**	27	26	12	13	4	8
FCFS	**Avg (s)**	**49582**	49595	49671	49671	49854	49932	50486	50486
	Dev (%)	**0.18%**	0.25%	0.68%	0.68%	1.54%	2.35%	5.49%	5.49%
	#Best	**97**	89	72	72	52	51	18	18

Table C2. Behavior in terms of completion time of different batching algorithms (GVNS, ILS, and FCFS) combined with S-Shape routing strategy and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).

		Completion time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
GVNS	**Avg (s)**	46911	46268	45706	45728	45662	**45627**	45670	45638
	Dev (%)	6.14%	3.56%	1.18%	1.23%	0.85%	0.73%	0.74%	**0.67%**
	#Best	1	6	12	15	28	**32**	22	29
ILS	**Avg (s)**	49303	48648	48055	48022	47940	47948	47968	**47934**
	Dev (%)	6.61%	4.08%	1.80%	1.71%	1.11%	**1.08%**	1.14%	1.12%
	#Best	2	5	17	14	**35**	20	26	30
FCFS	**Avg (s)**	52006	51401	50818	50818	50683	**50669**	50734	50734
	Dev (%)	6.06%	3.65%	1.24%	1.24%	0.59%	**0.48%**	0.64%	0.64%
	#Best	0	5	16	19	60	**61**	53	41

Table C3. Behavior in terms of maximum turnover time of different batching algorithms (GVNS, ILS, and FCFS) combined with S-Shape routing strategy and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH).

		Maximum turnover time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
GVNS	**Avg (s)**	43054	41922	40612	40848	40485	**40099**	40104	40154
	Dev (%)	72.49%	57.71%	29.79%	29.60%	13.00%	10.71%	9.90%	**8.28%**
	#Best	6	5	16	15	18	**34**	23	29
ILS	**Avg (s)**	45231	44122	43062	42963	42150	42380	**42027**	42093
	Dev (%)	68.20%	45.85%	30.75%	28.84%	9.52%	9.58%	**6.92%**	8.12%
	#Best	14	16	18	15	18	20	24	**27**
FCFS	**Avg (s)**	46737	45408	44108	44108	**43923**	43972	44053	44068
	Dev (%)	65.41%	36.62%	10.82%	10.82%	8.05%	7.64%	**5.71%**	5.76%
	#Best	2	9	22	21	**53**	44	**53**	48

Appendix D. Detailed results of the influence of time-window algorithms on the performance of the studied methods for different objective functions, when varying the number of pickers

Table D1. Behavior in terms of picking time of a GVNS batching algorithm, combined with S-Shape routing strategy and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) when varying the number of pickers.

		Picking time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
1 Picker	**Avg (s)**	**44017**	44071	44277	44295	44564	44611	45152	45115
	Dev (%)	**0.20%**	0.64%	1.95%	2.03%	3.57%	4.33%	7.65%	7.57%
	#Best	**76**	41	9	5	5	5	3	4
2 Pickers	**Avg (s)**	**44498**	44726	45272	45240	46096	47154	48824	48904
	Dev (%)	**0.15%**	1.26%	3.95%	3.79%	8.17%	15.47%	26.09%	26.25%
	#Best	**110**	23	5	5	0	0	2	0
3 Pickers	**Avg (s)**	**44924**	45275	46169	46121	47946	50646	53877	54237
	Dev (%)	**0.15%**	1.57%	5.36%	5.20%	12.76%	27.38%	45.96%	46.69%
	#Best	**108**	21	3	6	0	2	6	1
4 Pickers	**Avg (s)**	**45174**	45604	46822	46797	49531	54327	58518	59401
	Dev (%)	**0.16%**	1.77%	6.30%	6.10%	16.28%	39.18%	62.00%	64.06%
	#Best	**117**	11	5	1	0	1	9	0
5 Pickers	**Avg (s)**	**45220**	45773	47074	47043	50588	57319	62291	63918
	Dev (%)	**0.15%**	1.91%	6.56%	6.60%	18.53%	47.80%	74.45%	78.13%
	#Best	**118**	14	1	2	1	0	9	0

Table D2. Behavior in terms of completion time of a GVNS batching algorithm, combined with S-Shape routing strategy and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) when varying the number of pickers.

		Completion time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
1 Picker	**Avg (s)**	46911	46268	45706	45728	45662	**45627**	45670	45638
	Dev (%)	6.14%	3.56%	1.18%	1.23%	0.85%	0.73%	0.74%	**0.67%**
	#Best	1	6	12	15	28	**32**	22	29
2 Pickers	**Avg (s)**	27061	26501	26095	26125	25954	25869	25840	**25819**
	Dev (%)	8.07%	4.98%	2.66%	2.80%	1.75%	1.24%	1.02%	**0.83%**
	#Best	1	1	12	10	32	28	**34**	**34**
3 Pickers	**Avg (s)**	21568	21169	20889	20874	20743	20549	20555	**20505**
	Dev (%)	7.58%	5.22%	3.52%	3.49%	2.56%	1.35%	1.17%	**0.89%**
	#Best	0	1	9	7	18	**48**	47	40
4 Pickers	**Avg (s)**	19334	19113	18837	18810	18625	18463	18431	**18309**
	Dev (%)	7.02%	5.79%	4.04%	3.90%	2.61%	1.58%	1.13%	**0.54%**
	#Best	2	0	4	9	23	32	49	**58**
5 Pickers	**Avg (s)**	18141	18003	17748	17721	17522	17371	17256	**17181**
	Dev (%)	6.72%	5.87%	4.28%	4.13%	2.76%	1.71%	0.84%	**0.45%**
	#Best	1	0	5	5	21	29	**65**	59

Table D3. Behavior in terms of maximum turnover time of a GVNS batching algorithm, combined with S-Shape routing strategy and different time-window algorithms (VTW_B1, VTW_B2, VTW_B3, VTW_HE, VTW_M1, VTW_M2, FTW_NW, and FTW_ZH) when varying the number of pickers.

		Maximum turnover time							
		VTW_B3	**VTW_B2**	**VTW_B1**	**VTW_HE**	**VTW_M1**	**VTW_M2**	**FTW_ZH**	**FTW_NW**
1 Picker	**Avg (s)**	43054	41922	40612	40848	40485	**40099**	40104	40154
	Dev (%)	72.49%	57.71%	29.79%	29.60%	13.00%	10.71%	9.90%	**8.28%**
	#Best	6	5	16	15	18	**34**	23	29
2 Pickers	**Avg (s)**	22547	20602	18766	18843	17451	17388	17310	**17208**
	Dev (%)	226.61%	154.18%	93.41%	95.33%	25.49%	19.86%	11.67%	**10.54%**
	#Best	5	2	10	14	19	20	37	**51**
3 Pickers	**Avg (s)**	16200	14191	12055	12199	10259	10175	10254	**9910**
	Dev (%)	354.37%	249.60%	138.32%	145.16%	32.72%	25.01%	15.95%	**6.14%**
	#Best	1	4	7	6	20	23	49	**62**
4 Pickers	**Avg (s)**	13649	11733	9390	9543	7594	7248	7415	**6857**
	Dev (%)	421.39%	313.18%	175.75%	177.23%	43.66%	32.19%	31.63%	**4.41%**
	#Best	4	1	4	5	16	16	65	**75**
5 Pickers	**Avg (s)**	12333	10667	8155	8115	6017	5798	5818	**5358**
	Dev (%)	479.54%	379.97%	204.43%	199.38%	51.45%	37.71%	35.19%	**3.15%**
	#Best	1	1	3	2	11	21	**81**	78

Appendix E. Characteristics of the instances

	Dataset #1 (Albareda-Sambola et al. (2009))				Dataset #2 (Henn (2012))
	W1	**W2**	**W3**	**W4**	**W5**
Storage policy	Random / ABC				Random / ABC
Depot position	Center / Left corner				Center
Order size	U(1,7)	U(2,10)	U(5,25)	U(1,36)	U(5,25)
Item weight	1	1	1	U(1,3)	1
Batch capacity (weight)	12	24	150	80	30 / 45 / 60 / 75
# parallel aisles	4	10	25	12	10
# of picking positions per aisle	2x30	2x20	2x25	2x16	2x45
# of total picking positions	240	400	1250	384	900
Parallel aisle length	50 m	10 m	50 m	80 m	45 m
Parallel aisle width	4.3 m	2.4 m	5 m	15 m	5 m
# of instances	20	20	20	20	64
Travel speed (m/min.)	48	48	48	48	48
Extraction speed (items/min)	6	6	6	6	6
Batch setup time	3 min	3 min	3 min	3 min	3 min

Table E1. Characteristics of the warehouses and the work parameters.

	#Customer orders							
	40	**50**	**60**	**80**	**100**	**150**	**200**	**250**
4 Hours	0.167	0.208	0.250	0.334	0.417	0.625	0.834	1.042

Table E2. λ values for each number of orders.

Appendix F. Exact results

	Picking Time		Completion Time		Turnover Time	
	n_{max}	CPU Time (s)	n_{max}	CPU Time (s)	n_{max}	CPU Time (s)
H_abc1_40_29	13	2649	14	1325	13	1068
H_abc1_60_38	11	1546	16	2854	17	2840
H_abc1_80_63	12	589	16	214	17	3246
H_abc2_40_9	13	56	14	2163	13	2006
H_abc2_60_18	11	1900	13	2392	16	666
H_abc2_80_47	13	768	15	327	16	2849
H_ran1_40_31	12	3088	21	423	19	2195
H_ran1_60_38	11	581	12	1191	13	1718
H_ran1_80_63	12	1216	15	2165	16	1294
H_ran2_40_12	14	3222	28	19	19	3051
H_ran2_60_18	11	2501	12	559	15	2855
H_ran2_80_47	12	1025	15	3294	15	1653
A_W1_50_000	13	725	33	1274	25	3555
A_W1_100_030	13	1175	13	910	16	3399
A_W1_200_090	12	290	13	1901	14	3325
A_W2_150_060	12	1250	15	2583	14	1897
A_W2_200_090	14	319	21	77	17	1586
A_W2_250_000	12	488	12	751	12	530
A_W3_50_030	23	733	18	1936	15	2392
A_W3_150_060	16	3365	15	956	15	1010
A_W3_250_090	21	3600	20	1938	15	2715
A_W4_100_030	12	225	12	535	11	1275
A_W4_150_030	12	206	12	1242	11	3263
A_W4_200_060	11	2353	11	1199	11	1837

Table F1. Largest number of orders (n_{max}) per instance and objective function, that the exact model is able to handle within a time limit of one hour.