WORKING PAPER SERIES

Order picking with multiple pickers and due dates – Simultaneous solution of order batching, batch assignment and sequencing, and picker routing problems

André Scholz/Daniel Schubert/Gerhard Wäscher

Working Paper No. 5/2016

OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

FACULTY OF ECONOMICS
AND MANAGEMENT

Impressum (§ 5 TMG)

Herausgeber:
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Der Dekan

Verantwortlich für diese Ausgabe:
André Scholz/Daniel Schubert/Gerhard Wäscher
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Postfach 4120
39016 Magdeburg
Germany

http://www.fww.ovgu.de/femm

Bezug über den Herausgeber
ISSN 1615-4274

Order picking with multiple pickers and due dates – Simultaneous solution of order batching, batch assignment and sequencing, and picker routing problems

A. Scholz, D. Schubert, G. Wäscher

Abstract

In manual picker-to-parts order picking systems of the kind considered in this article, human operators (order pickers) walk or ride through the warehouse, retrieving items from their storage location in order to satisfy a given demand specified by customer orders. Each customer order is characterized by a certain due date until which all requested items included in the order are to be retrieved and brought to the depot. For the actual picking process, customer orders may be grouped (batched) into more substantial picking orders (batches). The items of a picking order are then collected on a picker tour through the warehouse. Thus, the picking process of each customer order in the batch is only completed when the picker returns to the depot after the last item of the batch has been picked. Whether and to which extend due dates are violated (tardiness) depends on how the customer orders are batched, how the batches are assigned to order pickers, how the assigned batches are sequenced and how the pickers are routed through the warehouse. Existing literature has only treated special aspects of this problem (i.e. the batching problem or the routing problem) so far. In this paper, for the first time, an approach is proposed which considers all aspects simultaneously. A mathematical model of the problem is introduced that allows for solving small problem instances in reasonable computing times. For larger instances, a variable neighborhood descent (VND) algorithm is presented which includes various neighborhood structures regarding the batching and sequencing problem. Furthermore, two sophisticated routing algorithms are integrated into the VND algorithm. By means of numerical experiments, it is shown that this algorithm provides solutions of excellent quality.

Keywords: Order Picking, Order Batching, Batch Sequencing, Picker Routing, Traveling Salesman, Variable Neighborhood Descent

Corresponding author:
André Scholz
Otto-von-Guericke University Magdeburg, Faculty of Economics and Management
Postbox 4120, 39016 Magdeburg, Germany
Phone: +49 391 67 11841
Fax: +49 391 67 18223
Email: andre.scholz@ovgu.de

1 Introduction

Order picking is a function which is critical for managing and operating distribution warehouses efficiently. It deals with the retrieval of items requested by external or internal customers (Petersen & Schmenner, 1999; Wäscher, 2004). In picker-to-parts systems, which are referred to in this paper, human operators (order pickers) walk or ride through the warehouse and collect the requested items from the different storage locations (Wäscher, 2004).

The items specified by a customer order usually have to be provided by a certain due date (Henn & Schmid, 2013). Negligence of due dates may delay subsequent shipment and/or production processes, and, as a consequence, results in an unacceptable customer satisfaction and high costs. Whether or to what extend due dates of a set of customer orders (wave) can be met is dependent on (1) how the customer orders are grouped into picking orders (Order Batching Problem), (2) how the picking orders are assigned to and sequenced by the order pickers (Batch Assignment and Sequencing Problem), and (3) how each order picker is routed in order to collect the items of each picking order (Picker Routing Problem). These problems are closely interrelated and, thus, should be solved simultaneously in order to provide solutions which comply with the given due dates in the best possible way. Literature dealing with solution approaches which explicitly take into account these problems simultaneously is almost non-existing. To the best of our knowledge, Chen et al. (2015) represent the only exception. Their approach is related to a problem environment more specific than the one considered in this paper. Furthermore, computing times become a critical issue and the numerical experiments demonstrate that this approach can only be applied to very small problem instances.

Consequently, in this paper we present a new, more competitive approach to what is called hereafter the Joint Order Batching, Assignment, Sequencing and Routing Problem (JOBASRP) and which includes an integrative view of the problems sketched above. We propose a mathematical programming formulation to this problem whose size increases polynomially with the number of customer orders. This model provides insights into the problem, but is only appropriate for solving small problem instances. Therefore, we also introduce a heuristic solution approach, namely a variable neighborhood descent algorithm, which incorporates neighborhood structures regarding the batching and the sequencing problem proposed in an earlier paper by Henn (2015). The arising routing problems are solved by means of the combined heuristic, which constructs routes of good quality within fractions of a second (Roodbergen & de Koster, 2001b). Furthermore, in order to improve the routes, the Lin-Kernighan-Helsgaun heuristic (Helsgaun, 2000) is applied to very promising solutions. By means of numerical experiments, it is shown that this approach leads to high-quality solutions within reasonable computing times even for large problem instances.

The remainder of this paper is organized as follows: In section 2, we give a precise statement of the JOBASRP. Section 3 comprises a literature review regarding the batching, the joint batching and sequencing, the routing, and the joint batching, sequencing and routing problem. For the latter problem, a new mathematical model formulation is presented in Section 4. Section 5 contains the description of the variable neighborhood descent algorithm including the generation of an initial solution, the different

neighborhood structures and the integration of the routing algorithms. In Section 6, the numerical experiments are presented which have been carried out in order to evaluate the algorithm. It is explained how the problem classes have been chosen, and the results from the experiments are discussed. The paper concludes with an outlook on further research.

2 Problem description

In the following, a warehouse with a manual, low-level picker-to-parts order picking system is considered from which a given set of items has to be retrieved. In low-level picker-to-parts systems, the items are stored on pallets or in bins directly accessible to the order pickers (Henn et al., 2012). The storage locations of the items typically constitute a block layout (Roodbergen, 2001) composed of so-called picking aisles and cross aisles. The picking aisles run parallel to each other and include storage locations arranged on both sides of each picking aisle. Cross aisles do not contain any storage locations, but enable order pickers to enter or exit a picking aisle. Furthermore, the cross aisles divide the picking area into several blocks and the picking aisles into subaisles. A block is formed by the picking area located between two adjacent cross aisles. The corresponding part of a picking aisle is denoted as a subaisle. Thus, a warehouse with m picking aisles and $q+1$ cross aisles includes q blocks and $q \cdot m$ subaisles. Additionally, the warehouse contains a depot where the order pickers enter the picking area and return to in order to deposit the picked items. In Fig. 1, an example of a picking area with two blocks and five picking aisles is depicted. A two block layout is characterized by three cross aisles, namely the front, middle and rear cross aisle, where the front and the rear cross aisles represent the cross aisles nearest to and farthest away from the depot, respectively. The middle cross aisle separates the two blocks from each other. The storage locations are represented by rectangles, while the black rectangles symbolize the locations of requested items (pick locations).

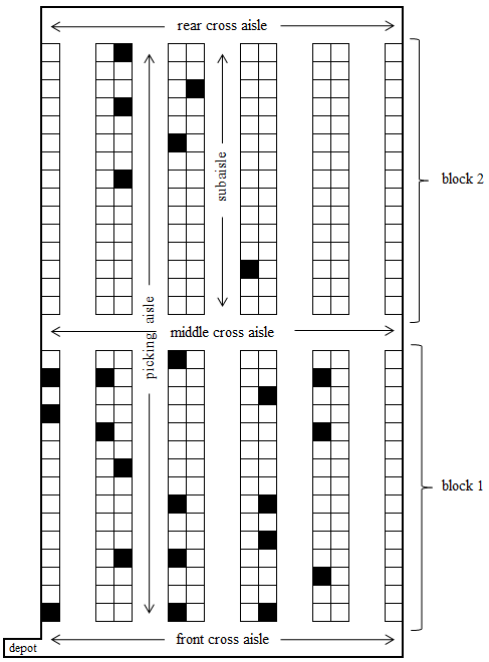

Fig. 1: Two-block layout

In order to retrieve the requested items, several order pickers operate in the picking area. Each order picker is equipped with a picking device, e.g. a cart or a roll cage, enabling him to perform a tour through the warehouse on which several items are picked before he returns to the depot. The maximum number of stops on a tour is dependent on the capacity of the picking device and the capacity requirements of the respective items to be picked. On his tour the order picker is guided by a so-called pick list. The list represents a batch and identifies the storage locations and the quantities of the items which are to be retrieved on the same tour. A batch may include requested items of several customer orders. However, splitting of customer orders is not allowed since it would result in an unacceptable sorting effort. The pick list also contains information on the sequence according to which the picker is meant to visit the respective pick locations.

The time an order picker spends retrieving all items of a batch (batch processing time) can be divided into (Tompkins et al., 2010):

- the setup time, i.e. the time needed for preparing a tour,
- the search time, i.e. the time required at each pick location for identifying the correct item,
- the pick time, i.e. the time for physically retrieving the items from their storage locations, and
- the travel time, i.e. the time spent for traveling from the depot to the first pick location, between the pick locations and back, and from the last pick location back to the depot.

Processing of a batch is started when the corresponding order picker to whom the batch has been assigned leaves the depot. This point in time is addressed as the batch start time, while the batch completion time denotes the point in time when the picker returns to the depot after having retrieved all items included in the batch. The start time (likewise: the completion time) of a customer order is defined as the start time (completion time) of the batch in which the order is included.

In practice, customer orders have to be completed by certain due dates (Henn & Schmid, 2013). In distribution warehouses, complying with the due dates is pivotal in order to guarantee the scheduled departure of trucks delivering the items to external customers (Gademann et al., 2001). We, therefore, evaluate the quality of a solution by means of the tardiness of the customer orders (Elsayed et al., 1993). The tardiness of a customer order is defined as the (nonnegative) difference between the completion time of the order and its due date (Henn & Schmid, 2013). The sum of the tardiness of all customer orders (total tardiness) specifies to which extent due dates are violated. A total tardiness of 0 means that the due dates of all customer orders are met in the respective solution.

The Joint Order Batching, Assignment, Sequencing and Picker Routing Problem (JOBASRP) can now be stated as follows: Let a non-empty set of customer orders be given, each of which including certain items with known storage locations to be removed from the warehouse. Furthermore, each order is characterized by a due date until which all requested items of the order should be retrieved and brought forward to the depot. For this purpose, a given number of order pickers is available. Then, the following questions have to be answered (simultaneously) in such a way that the total tardiness is minimized:

- How should the set of customer orders be grouped into picking orders? (Order Batching Problem)

- How and in which sequence should the set of picking orders be assigned to the order pickers? (Batch Assignment and Sequencing Problem)
- For each picking order, in which sequence should the respective pick locations be visited? (Picker Routing Problem)

In the following, we will assume a warehouse with wide aisles which enables the order pickers to overtake each other. This assumption has also been made by Chen et al. (2015) and Henn (2015) and allows for neglecting cases in which the processing time of a batch is increased by waiting times which may arise if order pickers simultaneously work in the same picking aisle. Even without considering such picker blocking aspects, the JOBASRP formulated above is known to be NP-hard (Chen et al., 2015).

3 Literature Review

Although the above-mentioned subproblems of the JOBASRP arise simultaneously, joint solution approaches have rarely been addressed in the literature so far. Instead, the subproblems are dealt with independent of each other in most approaches. This first subproblem is the Order Batching Problem (OBP) which can be stated as follows (Wäscher, 2004): Given the article storage locations, the routing strategy to be used, and the capacity of the picking device, how can the set of customer orders be grouped into picking orders such that the total lengths of all tours required to retrieve the items is minimized?

Since the OBP is known to be NP-hard if the number of orders per batch is greater than two (Gademann & van de Velde, 2005), only few exact solution approaches exist to the OBP. These approaches are based on integer programming, while Gademann & van de Velde (2005) and Muter & Öncan (2015) used a branch-and-price algorithm to solve OBPs with an arbitrary routing strategy, and Bozer & Kile (2008) and Öncan (2015) developed model formulations for certain routing strategies, respectively. For the heuristic solution of the OBP, many approaches have been proposed. Among constructive algorithms, priority rule-based (Gibson & Sharp, 1992), seed (Elsayed, 1981) and savings (Clarke & Wright, 1964) algorithms are the most prominent ones. Apart from these constructive approaches, several metaheuristics have been designed for the OBP which result in high-quality solutions within a reasonable amount of computing time. For a very detailed review of solution approaches to the OBP we refer to de Koster et al. (2007) and Henn et al. (2012).

The Batch Assignment and Sequencing Problem (BASP) is characterized by the fact that for each customer order due dates are given which have to be met in the best possible way, while each customer order has already been assigned to a certain batch. Solutions are evaluated by the total tardiness of all orders. The BASP can then be stated as follows: How should the batches be assigned to a limited number of pickers and, for each picker, how should the batches be sequenced such that the total tardiness is minimized?

The special case of the BASP in which each batch includes either only one customer order or customer orders with the same due date is equivalent to the Parallel Machine Scheduling Problem with the

objective of minimizing the total tardiness. In this problem, a set of jobs (here: batches) have to be assigned to machines (here: order pickers) and put in a sequence in such a way that the total tardiness of all jobs is minimized (Pinedo, 2016). We refer to Koulamas (1994) for a review of solution approaches to the Parallel Machine Scheduling Problem.

To the best of our knowledge, no solution approach exists to the BASP when the batches consist of customer orders with different due dates. However, a variety of algorithms has been proposed for the Joint Order Batching, Assignment and Sequencing Problem (JOBASP) which deals with the following questions (Henn, 2015): How should a given set of customer orders be grouped into batches, how should the batches be assigned to a limited number of pickers and how, for each picker, should the batches be sequenced such that the total tardiness is minimized?

Elsayed et al. (1993) considered the JOBASP with a single order picker in an automated storage and retrieval system (AS/RS). The proposed algorithm aims at a minimization of the total earliness and tardiness and can be divided into three steps. First, a priority value is assigned to each customer order dependent on its due date and the processing time required. Customer orders are then sequenced in a non-descending order according to their priority values. If a swap of two adjacent orders would result in an improved solution, the sequence will be changed accordingly. In step 2, batches are created. For each customer order, it is checked whether a combination with another order would improve the solution. If this is the case, the respective orders will be grouped into a batch. Finally, in the third step, the start time is determined for each batch.

Elsayed & Lee (1996) designed a solution approach to the JOBASP with a single picker in an AS/RS which aims at a minimization of the total tardiness. An initial solution is constructed by sorting the orders according to their due dates and scheduling them according to the position in the sorted list. Each order is processed on a separate tour. In order to improve the solution, orders are grouped into batches in the following way. First, a seed order is determined by means of the nearest schedule rule and assigned to an empty batch. Customer orders in the sequence are then added to the batch if this does not increase the total tardiness and does not violate the capacity constraint. When no further orders can be added, the next order in the sequence will serve as a seed order.

Henn & Schmid (2013) dealt with the same problem type but considered a manual order picking system. They proposed an iterated local search and an attribute-based hill climber algorithm. The authors compared the solution quality of their approaches to the quality of solutions constructed by the earliest due date rule in which customer orders are batched and sequenced according to their due dates. Henn & Schmid (2013) considered problem instances with up to 80 customer orders in their numerical experiments and demonstrated that the total tardiness can be reduced by 45% on average. The maximum computing time of the iterated local search and the attribute-based hill climber algorithm amounted to 10 minutes.

Henn (2015) extended the analysis to the case of multiple pickers. He proposed a variable neighborhood descent (VND) and a variable neighborhood search (VNS) approach to the JOBASP. As a benchmark, he modified the earliest due date rule in such a way that a batch is assigned to the picker which currently

possesses the smallest total processing time. Problem instances with up to 200 customer orders have been included in the numerical experiments. The total tardiness obtained by using the modified earliest due date rule could be reduced by 41% (VNS) and 39% (VND) on average. Application of the VNS algorithm requires up to 25 minutes of computing time, while the VND approach terminates after a maximum of 30 seconds.

The Picker Routing Problem (PRP) represents the third subproblem of the JOBASRP and can be stated as follows (Ratliff & Rosenthal, 1983; Scholz et al., 2016): Given a set of items to be picked from known storage locations, in which sequence should the locations be visited such that the total length of the corresponding picker tour is minimized? The PRP is a special case of the well-known Traveling Salesman Problem (TSP). However, with respect to the special structure of the arrangement of the storage locations in a warehouse, more specific solution approaches to the PRP have been suggested in the literature.

For the PRP in a single-block Ratliff & Rosenthal (1983) provided an efficient exact solution approach which was extended by Roodbergen & de Koster (2001a) to the case of a two-block layout. However, it can be seen that this algorithm is far more complex, and they stated that it would be very difficult to further extend their algorithm to deal with warehouses with more than two blocks. An exact solution approach to the PRP in warehouses with an arbitrary number of blocks has been presented by Scholz et al. (2016). Based on an observation of Burkard et al. (1998), the authors interpreted the PRP as a Steiner TSP and provided a specific graph for the representation of the problem. Application of a TSP-based model formulation resulted in a mathematical model to the PRP whose size is independent of the number of pick locations.

However, in practice, for the routing of order pickers usually simple strategies are applied which can be considered as heuristic solution approaches to the PRP. This is due to the fact that optimal tours appear to be complex and difficult to memorize for the order pickers. The S-shape, the return and the largest gap strategies are the simplest routing strategies (de Koster et al., 2007; Gu et al., 2007). The S-shape strategy is the most frequently used routing scheme in practice (Roodbergen, 2001). When applying this strategy, the order picker traverses each subaisle completely which contains at least one requested item. An exception may occur in the last subaisle of the block which is visited. Here, the picker may return after retrieving all items in this subaisle. A more sophisticated strategy, the so-called combined strategy (Roodbergen & de Koster, 2001b), combines elements of the S-shape and the return strategy. For each picking aisle to be visited, by means of dynamic programming it is determined whether the aisle is traversed or whether it is entered and left via the same cross aisle.

The solution quality of the routing strategies is strongly dependent on the problem data (number of subaisles, number of pick locations) and tours generated by these strategies may be far from optimal (Roodbergen, 2001). Theys et al. (2010) have shown that the tour length can be reduced by up to 48% by applying the Lin-Kernighan-Helsgaun heuristic (Helsgaun, 2000) instead of using the simple S-shape strategy. Furthermore, the advantage of simplicity diminishes when more complex warehouse layouts are considered as the routing strategies may also lead to quite confusing tours (Roodbergen, 2001). Therefore, in recent years, the integration of the PRP into other problems, in particular into the OBP, has

become more popular which gives rise to the Joint Order Batching and Picker Routing Problem (Scholz & Wäscher, 2015): Given a set of customer orders, each of which requiring certain items with known storage locations to be retrieved, how should the customer orders be grouped into picking orders and, for each picking order, in which sequence should the pick locations be visited such that the length of all tours is minimized?

Kulak et al. (2012) provided a mixed-integer programming formulation based on formulations for the Bin Packing and the Traveling Salesman Problem. Furthermore, they introduced a tabu search algorithm for the OBP and integrated two TSP heuristics in order to solve the arising PRPs. Another approach to this problem was presented by Grosse et al. (2014) who designed a simulated annealing algorithm for the batching problem and combined it with four different routing heuristics which were also used to create initial batches. Scholz & Wäscher (2015) proposed an iterated local search approach for the OBP and integrated both an exact and several heuristic approaches to the PRP. They demonstrated that – by taking the routing problem into account instead of using a fixed routing strategy – the total tour length can be decreased by up to 25% without increasing computing times.

As a consequence of a reduced total tour length also the total travel time that the order pickers have to spend in the warehouse for collecting the requested items will be reduced, resulting in shorter batch processing times which are pivotal for meeting due dates. As a preliminary conclusion from the review of the existing literature on order picking, it can thus be stated that a joint consideration of the JOBASP and the PRP would provide an excellent starting point for an improved planning of picking operations. However, only very few solution approaches to the JOBASRP exist.

Tsai et al. (2008) proposed a genetic algorithm for the JOBASRP. Besides the total tardiness, they also minimize the total earliness as well as the total tour length. Unlike in the approaches discussed before, splitting of customer orders is allowed. Due to this specific property, this approach is not considered any further.

Chen et al. (2015) also dealt with a special case of the JOBASRP as it has been defined in the previous section. The number of pickers is limited to one in their approach implying that the assignment problem does not have to be taken into account. The authors designed a genetic algorithm for the JOBASP and solved the arising PRPs by means of an ant colony approach. In the genetic algorithm, however, the batching and the sequencing problem are considered separately which results in many unfeasible solutions with respect to the capacity constraint of the picking device. Furthermore, the ant algorithm consumes far more computing time than problem-specific approaches to the PRP. Thus, it is not surprising the author stated that computing times are a critical issue. In their numerical experiments, only very small problem instances with up to 8 orders have been considered. Computing times have not been reported.

We conclude that no approach to the JOBASRP is available which can deal with larger problem instances within a reasonable amount of computing time. Furthermore, the case of multiple order pickers working simultaneously is rarely taken into account.

4 Model formulation

Henn (2015) introduced a mathematical model for the JOBASP which can be adapted to the JOBASRP. For this model, all feasible batches have to be generated in advance, where the number of feasible batches increases exponentially with the number of customer orders. For each feasible batch, the minimum processing time has then to be calculated, which means that the arising PRPs have to be solved to optimality. Thus, providing the problem data would consume a large amount of computing time. Furthermore, since the number of variables in the model depends on the number of feasible batches, even for small instances it may not be possible to generate the model due to memory restrictions.

Chen et al. (2015) also considered all feasible batches for their model, resulting in a similar magnitude of variables. Besides, some constraints of the model are nonlinear, making it even more difficult to solve.

We decided to choose a modeling approach in which the batches are not generated in advance in order to keep the number of variables at a reasonable level. The model formulation can be divided into two parts: The first part is related to the Joint Order Batching and Picker Routing Problem which means that the customer orders are grouped into batches and the corresponding tours are constructed. With respect to the specific structure of tours in an order picking warehouse, we interpret the routing problem as a Steiner TSP. The graph of Fig. 2 illustrates this interpretation and will be used as the basis for our model.

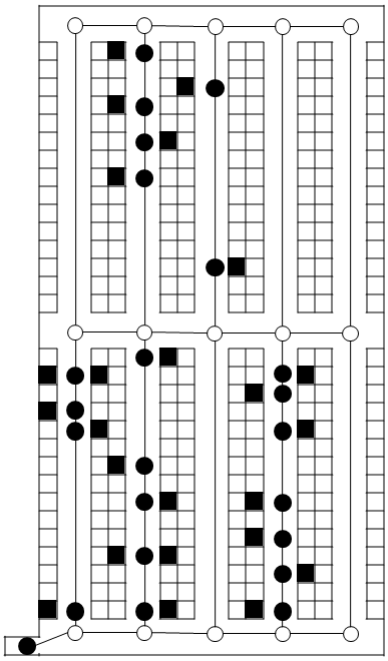

Fig. 2: Illustration of a Steiner TSP

The black vertices represent the location of the depot and the pick locations. These vertices have to be included in the tour. The white vertices (Steiner points) depict the intersections of picking aisles and cross aisles. They can be visited, but must not necessarily be visited. As can be taken from the figure, the maximum degree of a vertex is equal to four. Therefore, this Steiner TSP-based graph contains significantly fewer arcs than a standard TSP graph would include. In the model formulation, we consider a directed graph which means that each edge is replaced by two reverse arcs.

The second part of the model formulation deals with the Batch Sequencing Problem in which the constructed batches are assigned to a picker and arranged in a certain sequence. The sequence consists of as many positions as batches have been assigned to the picker. The picker starts processing the batch assigned to the first position and, after having returned to the depot, the batch in the next position is dealt with etc. This procedure provides the completion time for each batch and the completion times of all customer orders included in the batch, from which the tardiness of each customer order can be determined.

Before presenting the model formulation, we introduce the corresponding sets, parameters and variables.

Sets

P: set of order pickers

K: set of positions (for each order picker) where a batch can be scheduled $\left(K = \left\{1, \ldots, \overline{K}\right\}\right)$

H: set of tours which can be performed

N: set of customer orders

V: set of vertices in the graph representing the warehouse

R: set of vertices representing a pick location and the location of the depot

A: set of arcs in the graph representing the warehouse

Parameters

C: capacity of the picking device

β^s: setup time per batch

β^p: pick and search time per item

β^t: travel time per length unit

c_n: number of requested items of order $n \in N$

d_n: due date of order $n \in N$

α_{nr}: constant for indicating whether vertex $r \in R$ represents a storage location of a requested item of order $n \in N$ ($\alpha_{nr} = 1$) or not ($\alpha_{nr} = 0$)

w_a: distance to be covered when using arc $a \in A$

M: sufficiently large number $\left(\text{e.g. } M = \overline{K} \cdot \left(\beta^s + \beta^t \cdot \sum_{a \in A} w_a\right) + \beta^p \cdot \sum_{n \in N} c_n\right)$

Variables

τ_n: tardiness of order $n \in N$

$\tilde{u}_h$: processing time of tour $h \in H$

u_{pk}: processing time of the tour assigned to position $k \in K$ of picker $p \in P$

v_{pk}: completion time of the tour assigned to position $k \in K$ of picker $p \in P$

x_{pkh}: variable for indicating whether tour $h \in H$ is assigned to position $k \in K$ of picker $p \in P$ ($x_{pkh} = 1$) or not ($x_{pkh} = 0$)

y_{nh}: variable for indicating whether order $n \in N$ is assigned to tour $h \in H$ ($y_{nh} = 1$) or not ($y_{nh} = 0$)

z_{ah}: variable for indicating whether arc $a \in A$ is included in tour $h \in H$ ($z_{ah} = 1$) or not ($z_{ah} = 0$)

f_{ah}: number of units of the commodity passing arc $a \in A$ on tour $h \in H$

The JOBASRP can then be formulated as follows.

$$\min \sum_{n \in N} \tau_n \tag{1}$$

$$\sum_{h \in H} x_{pkh} \leq 1 \quad \forall p \in P,\, k \in K \tag{2}$$

$$\sum_{p \in P} \sum_{k \in K} x_{pkh} = 1 \quad \forall h \in H \tag{3}$$

$$\sum_{h \in H} y_{nh} = 1 \quad \forall n \in N \tag{4}$$

$$\sum_{n \in N} c_n \cdot y_{nh} \leq C \quad \forall h \in H \tag{5}$$

$$\sum_{a \in \delta_0^-} z_{ah} \geq 1 \quad \forall h \in H \tag{6}$$

$$\sum_{a \in \delta_r^-} z_{ah} \geq \alpha_{nr} \cdot y_{nh} \quad \forall h \in H,\, n \in N,\, r \in R \backslash \{0\} \tag{7}$$

$$\sum_{a \in \delta_v^-} z_{ah} = \sum_{a \in \delta_v^+} z_{ah} \quad \forall h \in H,\, v \in V \tag{8}$$

$$\sum_{a \in \delta_r^+} f_{ah} - \sum_{a \in \delta_r^-} f_{ah} = \alpha_{nr} \cdot y_{nh} \quad \forall h \in H,\, n \in N,\, r \in R \backslash \{0\} \tag{9}$$

$$\sum_{a \in \delta_v^+} f_{ah} - \sum_{a \in \delta_v^-} f_{ah} = 0 \quad \forall h \in H,\, v \in V \backslash R \tag{10}$$

$$f_{ah} \leq C \cdot z_{ah} \quad \forall a \in A,\, h \in H \tag{11}$$

$$\beta^s + \beta^p \cdot \sum_{n \in N} c_n \cdot y_{nh} + \beta^t \cdot \sum_{a \in A} w_a \cdot z_{ah} \leq \tilde{u}_h \quad \forall h \in H \tag{12}$$

$$\tilde{u}_h - M \cdot (1 - x_{pkh}) \leq u_{pk} \quad \forall p \in P,\, k \in K,\, h \in H \tag{13}$$

$$u_{p1} \leq v_{p1} \quad \forall p \in P \tag{14}$$

$$u_{pk} + v_{p,k-1} \leq v_{pk} \quad \forall p \in P,\, k \in K \backslash \{1\} \tag{15}$$

$$v_{pk} - d_n - M \cdot (2 - x_{pkh} - y_{nh}) \leq t_n \quad \forall n \in N,\, p \in P,\, k \in K,\, h \in H \tag{16}$$

$$x_{pkh},\, y_{nh},\, z_{ah} \in \{0, 1\} \quad \forall a \in A,\, n \in N,\, p \in P,\, k \in K,\, h \in H \tag{17}$$

$$\tau_n,\, \tilde{u}_h,\, u_{pk},\, v_{pk},\, f_{ah} \geq 0 \quad \forall a \in A,\, n \in N,\, p \in P,\, k \in K,\, h \in H \tag{18}$$

The objective function (1) minimizes the total tardiness. Constraints (2) ensure that at most one tour is assigned to each position of each picker, while (3) guarantee that each tour is performed. Each customer order has to be processed on exactly one tour which is obtained by meeting restrictions (4). The capacity

of the picking device is taken into account by satisfying (5). Constraints (6) to (11) represent the routing constraints. First, constraints (6) ensure that the depot is left on each tour. Here, δ_v^+ (δ_v^-) denotes the set of arcs to which vertex v is an end (start) vertex. Vertex "0" represents the location of the depot. Constraints (7) guarantee that each pick location is visited which corresponds to a requested item of a customer order included in the respective tour. Restrictions (8) are the degree constraints. The following three types of constraints represent subtour elimination constraints. These constraints are related to the single-commodity flow constraints introduced by Letchford et al. (2013). Subtours are excluded by ensuring that the picker starts his tour with a certain number of units of a commodity and delivers one unit to each vertex to be visited. In this way, the vertices are enumerated according to their appearance in the tour. In constraints (12), the processing time is determined for each tour, which is composed of the setup time, the time for searching and picking the items, and the travel time. The processing time of the tour assigned to a certain position of a certain picker is determined in (13), while constraints (14) and (15) calculate the corresponding completion times. Finally, constraints (16) compute the tardiness for each order. The variable domains are defined in (17) and (18).

The mathematical model includes linear constraints only. Furthermore, both the number of variables and the number of constraints increase polynomially with the problem size, which is a major advantage of this model in comparison to the formulations of Chen et al. (2015) and Henn (2015). However, our model formulation is not suitable for solving large problem instances as well, which provides the reason why we have developed a variable neighborhood descent approach to the JOBASRP.

5 Variable neighborhood descent

5.1 Overview

Variable neighborhood descent (VND) was introduced in Hansen & Mladenović (2001). The general principle of VND consists in exploring the solution space of the problem by means of a sequence of neighborhood structures $\mathcal{N}_1, \ldots, \mathcal{N}_L$. It is started with an incumbent solution s^* and the best neighbor s (in terms of the objective function value) of $\mathcal{N}_1(s^*)$ is determined. If s represents a better solution than s^*, then s becomes the new incumbent solution and the first neighborhood structure $\mathcal{N}_1(s^*)$ is considered again. Otherwise, the exploration of the solution space continues with the next neighborhood. The algorithm terminates if no improvement can be found in the last neighborhood structure $\mathcal{N}_L(s^*)$, i.e. a local optimum is found with respect to all neighborhood structures.

In our VND approach, a solution s is a solution to the JOBASP, i.e. it includes information about how the orders are grouped into batches and in which sequence the batches are to be processed by the pickers. Based on this solution, different PRPs have to be solved in order to determine the corresponding objective function values. Since a very large number of PRPs arise during the solution process, we decided to solve them by applying the combined heuristic. This heuristic was particularly designed for warehouses with multiple blocks and outperforms the frequently used S-shape heuristic by far in terms of solution quality (Roodbergen & de Koster, 2001b). The corresponding objective function value is denoted by $f_{\text{Comb}}(s)$.

In contrast to the standard VND procedure, we do not directly return to $\mathcal{N}_1$ after an improvement has been found. Instead, before doing so, we determine a local optimum regarding the current neighborhood structure. The objective function value, the total tardiness, is strongly dependent on the processing times of the batches. In order to reduce processing times, each time when a local optimum has been found, the arising PRPs are solved by means of the Lin-Kernighan-Helsgaun (LKH) heuristic. Applied to the PRP, this heuristic results in very short tours (Theys et al., 2010) and, as a consequence, in shorter processing times. However, since the computational effort is much higher than the effort for the application of the combined heuristic, it is not possible to use the LKH heuristic for evaluating all solutions to be considered in the exploration of the neighborhood structures. Due to this fact, we try to ensure that the LKH heuristic is only applied to very promising solutions. Therefore, we decided to determine a local optimum before returning to $\mathcal{N}_1$. $f_{\text{LKH}}(s)$ denotes the total tardiness of a solution s whose corresponding PRPs have been solved by means of the LKH heuristic. A general pseudocode of our VND approach is depicted below, while the generation of the initial solution as well as the different neighborhood structures will be dealt with in the next subsections.

Algorithm 1 Variable neighborhood descent algorithm for the JOBASRP

Input: problem data, number of neighborhood structures L

Output: solution s^* to the JOBASRP and corresponding total tardiness $f_{\text{LKH}}(s^*)$

```
generate initial solution s;
s* := s; l := 1;
while l ≤ L do
    s := s*;
    s' := arg min {f_Comb(s̃) | s̃ ∈ N_l(s)};
    while f_Comb(s') < f_Comb(s) do
        s := s';
        s' := arg min {f_Comb(s̃) | s̃ ∈ N_l(s)};
    endwhile
    if f_LKH(s) < f_LKH(s*) then
        s* := s;
        l := 1;
    else
        l := l + 1;
    endif
endwhile
```

5.2 Initial solution

For generating an initial solution, two constructive approaches are first applied, providing one solution each. The solution with the smaller objective function value is then taken as the initial solution for the VND. The first constructive approach is based on the earliest start date rule (ESDR) and was also used by Henn (2015). It is a priority rule-based algorithm in which orders are assigned successively to the batches and the positions of the pickers. A detailed pseudocode of this algorithm is depicted below.

Algorithm 2 ESDR-based algorithm

Input: set of orders N with due dates d_n and number of requested items c_n $(n \in N)$, set of pickers P, capacity of the picking device C

Output: solution s^* to the JOBASRP and corresponding total tardiness $f_{\text{Comb}}(s^*)$

$U := N$;
for $p \in P$ **do**
  $k_p := 1;\ B_{pk_p} := \emptyset;\ C_p := 0;\ v_p := 0;$
endfor
while $U \neq \emptyset$ **do**
  $n^* = \arg\min\{d_n \mid n \in U\}$;
  for $p \in P$ **do**
    if $C_p + c_{n^*} \leq C$ **then**
      $\tilde{v}_p := v_p + u_{\text{Comb}}\left(B_{pk_p} \cup \{n^*\}\right)$;
    else
      $\tilde{v}_p := v_p + u_{\text{Comb}}(\{n^*\})$;
    endif
  endfor
  $p^* := \arg\min\{\tilde{v}_p \mid p \in P\}$;
  $U := U \setminus \{n^*\};\ v_{p^*} := \tilde{v}_{p^*}$;
  if $C_{p^*} + c_{n^*} \leq C$ **then**
    $B_{p^*k_{p^*}} := B_{p^*k_{p^*}} \cup \{n^*\};\ C_{p^*} := C_{p^*} + c_{n^*}$;
  else
    $k_{p^*} := k_{p^*} + 1;\ B_{p^*k_p} := \{n^*\};\ C_{p^*} := c_{n^*}$;
  endif
endwhile

In the pseudocode, U denotes the set of orders not yet assigned to a picker, k_p is the sequence position of picker p to which the next batch can be assigned and B_{pk_p} represents the set of orders included in the batch assigned to position k_p of picker p. C_p and v_p denote the number of items contained in the batch under consideration for picker p as well as its completion time, while u_{Comb} calculates the processing time of the current batch by means of the combined heuristic. At the beginning of the algorithm, all orders are unassigned. The orders are then assigned successively to certain batches, starting with the unassigned order n^* with the smallest due date. For each order picker p, the completion time $\tilde{v}_p$ is determined that would follow from an assignment of n^* to the picker. The assignment consists of an addition of n^* to batch B_{pk_p} if the capacity constraint of the picking device is not violated; otherwise, the assignment includes the opening of a new batch containing n^*. The algorithm terminates when all orders have been assigned to batches and positions.

In order to meet the due dates and minimize the total tardiness, the orders are sorted according to their due dates and then assigned successively. The composition of the batches is not considered any further. This is a proper approach as long as the due dates are loose. However, in the case of tight due dates, it is of prime importance to construct batches and tours which allow for short processing times. We, therefore, propose

another constructive approach taking into account the processing times resulting from the construction of the batches. This approach can be perceived as a seed algorithm. Seed algorithms have been frequently applied to the OBP and consist of two steps. First, a seed order is chosen by means of a seed selection rule and assigned to a new batch. According to an order addition rule, orders are then added to this batch. As for the seed selection, we chose an order which is not yet assigned to a batch and has the closest due date. Dependent on the savings in terms of total tardiness, which result from adding an order to this batch instead of processing it separately, other orders are assigned to the batch. A detailed pseudocode of this algorithm can be seen below.

Algorithm 3 Seed algorithm

Input: set of orders N with due dates d_n and number of requested items c_n $(n \in N)$, set of pickers P, capacity of the picking device C

Output: solution s^* to the JOBASRP and corresponding total tardiness $f_{\text{Comb}}(s^*)$

$U := N;$ //step 1
for $p \in P$ **do**
  $k_p := 1;\ v_p := 0;$
endfor
while $U \neq \emptyset$ **do**
  $n^* := \arg\min \{d_n \mid n \in U\};$
  for $p \in P$ **do**
    $\tilde{v}_p := v_p + u_{\text{Comb}}(\{n^*\});$
  endfor
  $p^* := \arg\min \{\tilde{v}_p \mid p \in P\};$
  $U := U \setminus \{n^*\};\ v_{p^*} := \tilde{v}_{p^*};\ k_{p^*} := k_{p^*} + 1;\ B_{p^*k_p} := \{n^*\};$
endwhile
$U := N;\ i := 1;$ //step 2
while $U \neq \emptyset$ **do**
  $n^* := \arg\min \{d_n \mid n \in U\};\ \tilde{B}_i := \{n^*\};\ \tilde{C} := c_{n^*};\ U := U \setminus \{n^*\};$
  while $\max \left\{ sav_{in} \mid n \in U : \tilde{C} + c_n \leq C \right\} > 0$ **do**
    $n^* := \arg\max \left\{ sav_{in} \mid n \in U : \tilde{C} + c_n \leq C \right\};$
    $\tilde{B}_i := B_i \cup \{n^*\};\ \tilde{C} := \tilde{C} + c_n;\ U := U \setminus \{n^*\};$
  endwhile
  $i := i + 1;\ \tilde{C} := 0;$
endwhile
for $p \in P$ **do** //step 3
  $k_p := 1;$
endfor
for $j := 1$ to i **do**
  $(p^*, k^*) := \arg\min \left\{ f^{p,k}_{\text{Comb}} \mid p \in P, k \in \{1, \ldots k_p\} \right\};$
  $B_{p^*,k^*+1} := B_{p^*,k^*};\ ...;\ B_{p^*,k_p+1} := B_{p^*,k^p};\ B_{p^*,k^*} := \tilde{B}_i;\ k_{p^*} := k_{p^*} + 1;$
endfor

In the first step, each order is assigned to a separate batch. Starting with the batch that includes the order with the smallest due date, the batches are successively added to the picker p who currently possesses the shortest completion time $\tilde{v}_p$. In the second step, batches are merged in order to reduce the processing times as well as the total tardiness. The order with the smallest due date represents the seed order and forms batch $\tilde{B}_1$. Based on the solution generated in step 1, for each unassigned order $n \in U$, savings sav_{in} are determined which are defined as the reduction of the total tardiness obtained by merging the batch of order n and the batch i of the seed order. Of the potential pairs of batches which could be merged without violating the capacity constraint of the picking device, two batches are actually merged for which the savings are maximal. Then, the savings are updated and another order may be added to the batch. This is done until no further positive savings can be realized. While at least one order exists not considered in step 2, i. e. while $U \neq \emptyset$, the next seed order is determined based on the due dates and a new batch is opened to which orders are to be added. Thus, step 2 provides a solution in which the processing times and the total tardiness are explicitly taken into account. In the last step, the batches in this solution are reassigned to the position of the pickers. This is done consecutively starting with batch $\tilde{B}_1$. Each batch is inserted into the position k of picker p minimizing the total tardiness $f_{\text{Comb}}^{p,k}$ of all batches which have been inserted up to this point. In analogy to the ESDR-based algorithm, the combined heuristic is used to determine the processing time of a batch.

Since the seed algorithm also considers the composition of the batches instead of just basing the construction of a solution on the information about the due dates, it can be expected to lead to better results in the case of tight due dates. By selecting the best solution of the ESDR-based algorithm and the seed algorithm, we aim to generate a good initial solution that enables the VND to proceed faster to a local optimum.

5.3 Neighborhood structures

As mentioned before, a solution to the JOBASRP includes information about the composition of the batches and their assignment to the positions of the pickers. The arising routing problems are only solved in order to determine the resulting objective function value. Thus, the neighborhood structures considered in our VND can be divided into two categories, namely structures related to the sequencing problem and structures related to the batching problem.

Neighborhood structures related to the sequencing problem only deal with the assignment of the batches to the pickers' positions. The composition of the batches remains unchanged. Thus, complete batches are considered instead of single customer orders. Since the number of batches is usually much smaller than the number of customer orders, these neighborhoods can be explored in reasonable computing time. Therefore, these neighborhoods are used at the beginning of the VND algorithm. As proposed by Henn (2015), we use one sequencing related neighborhood structure $\mathcal{N}_1$.

Regarding $\mathcal{N}_1$, the neighborhood of a solution s is composed of all solutions which can be obtained by exchanging two batches b_1 and b_2 from s. Only exchanges between different pickers are taken into consideration. Batch b_1 is moved to the position of b_2 and vice versa. The positions of the remaining

batches are not changed (see Fig. 3).

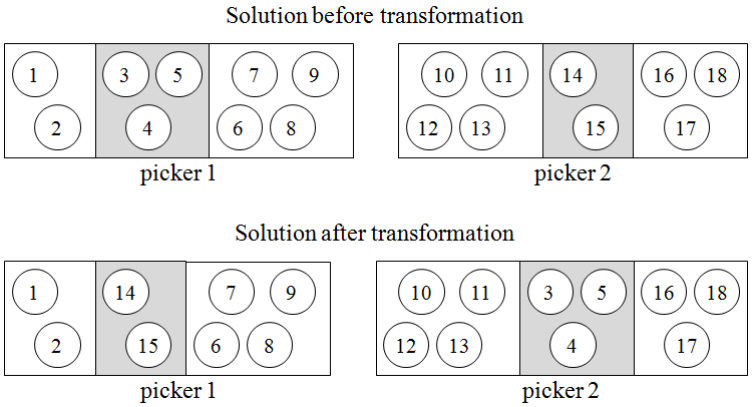

Fig. 3: Example of neighborhood structure $\mathcal{N}_1$

In $\mathcal{N}_2$, we consider complete batches, too. However, not only the way how batches are sequenced is changed but also how they are composed of orders. This neighborhood structure is meant to break up a complete batch and reassign the orders to other batches. Furthermore, the batch sequence is optimized. A neighbor regarding $\mathcal{N}_2$ is obtained as follows: In a first step, a batch is removed from the solution s. In a second step, the remaining sequence is optimized. First, for each batch, it is checked whether the solution can be improved by moving this batch to a position of another picker. The move which provides the largest improvement is carried out. This procedure is repeated until no further improvement is possible. Movements regarding $\mathcal{N}_1$ are then applied as long as the solution can be improved. Subsequent to this improvement step, the orders contained in the removed batch are considered successively in the order of non-descending due dates. Each order may either be inserted in an existing batch or forms a new batch. The option resulting in the smallest total tardiness is realized. An example of a move regarding $\mathcal{N}_2$ is depicted in Fig. 4.

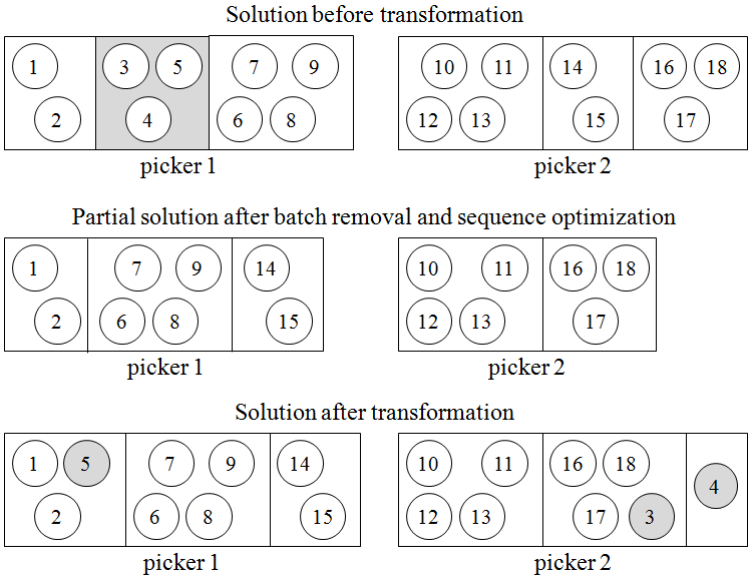

Fig. 4: Example of neighborhood structure $\mathcal{N}_2$

Neighborhood structures $\mathcal{N}_3$ to $\mathcal{N}_6$ are straightforward structures to the batching problem. By means of these structures, the composition of the batches is changed. Either an order is moved from one batch

to another batch (shift) or two orders contained in different batches are exchanged (swap). These structures can define movements on the same picker as well as operations including two pickers. Combination of these characteristics (swap or shift and one picker or two pickers) gives rise to four different neighborhood structures which have been used in the following sequence:

$\mathcal{N}_3$: an order is moved from one batch to another batch assigned to the same picker;
$\mathcal{N}_4$: an order is moved from one batch to another batch assigned to another picker;
$\mathcal{N}_5$: two orders are exchanged which are included in different batches assigned to the same picker;
$\mathcal{N}_6$: two orders are exchanged which are included in different batches assigned to different pickers.

We note that only neighbors are considered which represent feasible solutions, i.e. which do not violate the capacity constraint of the picking device. If an order cannot be added to a batch, this order is assigned to a new batch, where the position of the new batch is chosen in such a way that the total tardiness is minimized.

6 Numerical Experiments

6.1 Setup

The performance of the VND is evaluated in a series of numerical experiments. Except for the number of order pickers available, the problem definition of Chen et al. (2015) is identical with the one considered in this paper. Unfortunately, the problem instances from their experiments were not available. We thus generated instances with the same characteristics and noticed that all instances can be solved to optimality within a few seconds by means of the model formulation proposed in Section 4. We, therefore, decided to not use these instances for the evaluation of the solution quality of our VND. Instead, we adapted the numerical experiments of Henn (2015) who dealt with very large instances for the JOBSP.

In our experiments, we consider a warehouse with 10 picking aisles. The number of blocks q is varied between 1 and 3, resulting in 10, 20 or 30 subaisles. Each subaisle contains 50 storage locations (25 on each side of the subaisle). The depot is located in front of the leftmost picking aisle, and the distance between the depot and the leftmost picking aisle amounts to 1.5 length units (LU). For entering or leaving a subaisle, the order picker has to cover a distance of 1 LU. This is also the distance between two adjacent storage locations of a subaisle. The distance between two adjacent picking aisles amounts to 5 LU.

For the assignment of articles to storage locations, a class-based storage assignment policy is applied, i.e. articles with high demand frequencies are assigned to storage locations near the depot. We use the same approach as Henn (2015), who divided the articles into three classes A, B and C, whereupon class A includes 10% of all articles with the highest demand frequency, representing up to 52% of the total demand; class B contains 30% of all articles responsible for 36% of the demand. The remaining articles, assigned to class C, are characterized by rather low demand frequencies. Within each class, articles are randomly assigned to the storage locations of the corresponding subaisles. The determination of the subaisles is based on the distance to the depot. Class A articles are assigned to storage locations in a

subaisle representing 10% of all subaisles with the shortest distance to the depot. The subaisles which belong to 60% of all subaisles farthest from the depot include class C articles.

The number of customer orders is fixed to 100 and 200, whereupon the number of items contained in an order is uniformly distributed over the set $\{5, 6, \ldots, 25\}$. Each order should be completed before a certain due date. The due dates are generated based on the processing time $\tilde{u}_n$ $(n \in N)$ of the orders, the number of available order pickers $p_{\max}$ and the so-called modified traffic congestion rate (MTCR) γ describing the tightness of the due dates (Elsayed & Lee, 1996). The interval from which due dates are randomly chosen is determined as follows: $\left[\min\{\tilde{u}_n \mid n \in N\}, \left(2 \cdot (1-\gamma) \cdot \sum_{n \in N} \tilde{u}_n + \min\{\tilde{u}_n \mid n \in N\}\right) / p_{\max}\right]$. As can be seen, the tightness of the due dates is dependent on the processing time $\tilde{u}_n$ of an order n, while the processing time is determined by the sequence according to which the corresponding items are to be retrieved, which is the main reason why we do not use the same instances as Henn (2015). They applied the simple S-shape and largest gap strategies for the determination of the processing times. Since we integrate the routing problem into our approach, we generate much shorter tours. This results in considerable shorter batch processing times, which is why an application of our approach to the instances of Henn (2015) results in a total tardiness of 0 for most instances. Therefore, we decided to use the LKH heuristic for the determination of the processing times $\tilde{u}_n$ $(n \in N)$ instead, which is more appropriate as the LKH heuristic is also applied within our VND algorithm. The MTCR γ has been fixed to 0.6, 0.7 and 0.8 as done by Henn (2015).

In our experiments, 2, 3 or 5 pickers are available for processing the orders. The capacity C of the picking device is measured in the maximum number of items which can be picked on a tour and set to 45 or 75. The time a picker needs for performing a tour is composed of the setup time, the search and pick time, and the travel time (see Section 2). Here, the setup time amounts to 3 minutes, while searching and picking an item requires 20 seconds, and the order picker moves 20 LU per minute.

Combination of the above-mentioned parameter values results in 108 problem classes. For each problem class, 30 instances have been generated, i.e. 3240 problem instances have been solved in total. The experiments have been carried out on a desktop PC with a 3.4 GHz Pentium processor and 8 GB RAM. The solution approach has been encoded in C++ using Microsoft Visual Studio 2015.

6.2 Results

6.2.1 Evaluation of the initial solution

Pretests have shown that the amount of computing time required for applying the VND is strongly dependent on the quality of the initial solution. Starting with a low quality solution, a large number of iterations has to be carried out until a local optimum is found. Henn (2015) used the ESDR-based algorithm in order to generate an initial solution. However, this rule may result in very poor solutions since the orders are grouped into batches without considering the resulting processing times. Therefore, we proposed a seed algorithm in which both the due dates of the orders and the tour length of the corresponding batches are taken into consideration.

The computing times of both approaches are below one second if the combined heuristic is used for determining the processing times of the respective tours. Since such computing times can be neglected, both approaches are applied and the solution with the smaller total tardiness is used as the initial solution. The impact on the quality of the initial solution is depicted in Tables 1 and 2.

Table 1: Evaluation of the initial solution for 100 orders

			2 pickers			3 pickers			5 pickers		
C	γ	q	**ESDR**	**Initial**		**ESDR**	**Initial**		**ESDR**	**Initial**	
			tar	tar	imp	tar	tar	imp	tar	tar	imp
45	0.6	1	130	130	0.0	112	112	0.0	89	89	0.0
45	0.6	2	70	70	0.0	165	158	3.7	128	128	0.0
45	0.6	3	235	220	6.3	347	283	18.4	517	516	0.2
45	0.7	1	1831	1017	44.4	1451	1102	24.0	1209	986	18.4
45	0.7	2	2479	1345	45.8	2245	1708	23.9	1483	1280	13.7
45	0.7	3	5347	2715	49.2	3908	2435	37.7	2808	2147	23.5
45	0.8	1	8117	4988	38.6	6181	3996	35.4	4729	3294	30.3
45	0.8	2	10468	6522	37.7	7203	4734	34.3	6692	4502	32.7
45	0.8	3	14434	8968	37.9	10246	6677	34.8	3857	2694	30.2
75	0.6	1	62	62	0.0	85	85	0.0	116	116	0.0
75	0.6	2	63	63	0.0	99	99	0.0	130	130	0.0
75	0.6	3	91	91	0.0	154	154	0.0	270	270	0.0
75	0.7	1	141	141	0.0	166	166	0.0	268	268	0.0
75	0.7	2	181	181	0.0	340	340	0.0	312	312	0.0
75	0.7	3	431	431	0.0	489	489	0.0	686	686	0.0
75	0.8	1	2793	2473	11.5	2556	2399	6.1	2397	2250	6.1
75	0.8	2	3872	3407	12.0	2968	2742	7.6	3726	3134	15.9
75	0.8	3	6477	5120	21.0	5055	4289	15.1	1926	1778	7.7
average			3179	2108	16.9	2432	1776	13.4	1741	1366	9.9

In Tables 1 and 2, for the problem classes with 100 and 200 orders, the quality of the initial solutions (in terms of the average total tardiness (tar) in minutes) is depicted which is obtained after the application of the ESDR-based algorithm and after the additional application of the seed algorithm. The improvement (imp) amounts to zero, if the additional application of the seed algorithm has no impact on the quality of the initial solutions. In these cases, solutions constructed by means of the ESDR-based algorithm always lead to a smaller total tardiness for the instances of the corresponding problem class. In fact, this is the case for instances with a very small MTCR ($\gamma = 0.6$) and with a medium MTCR ($\gamma = 0.7$) and a large capacity of the picking device ($C = 75$). These instances are characterized by quite loose due dates which can be met easily. (Note that the generation of the due dates is independent of the capacity of the picking device and due dates can be satisfied easier when the capacity is large since the total processing time of all orders decreases.) If the due dates are not tight, minimizing the processing times gets much less important since the due dates of most orders can be met by processing the orders in the sequence of non-descending due dates. This is exactly what the ESDR-based algorithm guarantees, which is the reason why the application of this rule leads to rather good solutions in these cases.

With an increasing MTCR, the due dates get tighter and harder to meet, resulting in a dramatic

Table 2: Evaluation of the initial solution for 200 orders

			2 pickers			3 pickers			5 pickers		
C	γ	q	**ESDR**	**Initial**		**ESDR**	**Initial**		**ESDR**	**Initial**	
			tar	tar	imp	tar	tar	imp	tar	tar	imp
45	0.6	1	158	158	0.0	181	180	0.7	180	180	0.0
45	0.6	2	130	130	0.0	115	115	0.0	270	270	0.0
45	0.6	3	753	604	19.8	345	345	0.0	419	419	0.0
45	0.7	1	6436	2755	57.2	5127	3551	30.7	3187	2784	12.7
45	0.7	2	8150	3461	57.5	5564	4212	24.3	3625	3307	8.8
45	0.7	3	18697	6972	62.7	12763	7692	39.7	9461	6865	27.4
45	0.8	1	33890	19487	42.5	22982	14169	38.3	14632	9494	35.1
45	0.8	2	41291	24360	41.0	26747	16759	37.3	17909	11872	33.7
45	0.8	3	56178	32847	41.5	39085	23840	39.0	24405	15708	35.6
75	0.6	1	70	70	0.0	85	85	0.0	136	136	0.0
75	0.6	2	70	70	0.0	97	97	0.0	164	164	0.0
75	0.6	3	95	95	0.0	158	158	0.0	267	267	0.0
75	0.7	1	177	177	0.0	190	190	0.0	267	267	0.0
75	0.7	2	375	375	0.0	235	235	0.0	286	286	0.0
75	0.7	3	478	478	0.0	618	618	0.0	755	755	0.0
75	0.8	1	11039	8705	21.1	7982	7454	6.6	5785	5526	4.5
75	0.8	2	14669	11758	19.8	9334	8852	5.2	7471	7011	6.2
75	0.8	3	22468	16607	26.1	16942	13767	18.7	11703	9974	14.8
average			11951	7173	21.6	8253	5684	13.4	5607	4183	9.9

increase of the average total tardiness. In the case of very tight due dates, it is not sufficient to find a reasonable sequence according to which the orders are processed. Instead, minimizing the processing times gets pivotal. Since the processing times are not taken into consideration, solutions obtained by means of the ESDR-based algorithm are expected to be of very poor quality. The results from the numerical experiments confirm this expectation and demonstrate that the seed algorithm outperforms the ESDR-based algorithm by far when a high MTCR is assumed. By means of the seed algorithm, the average total tardiness can be decreased by up to 62.7% (200 orders, 2 pickers, $C = 45$, $\gamma = 0.7$, $q = 3$). As expected, the savings obtained by application of the seed algorithm tend to get larger with a decreasing capacity of the picking device or an increasing MTCR. For problem classes with a small capacity ($C = 45$) and a large MTCR ($\gamma = 0.8$), the reduction of the average total tardiness ranges between 30.2% (100 orders, 5 pickers, $q = 3$) and 42.5% (200 orders, 2 pickers, $q = 1$). Furthermore, it should be noted that the improvements get smaller with an increasing number of order pickers available. This can be deduced to the fact that dealing with the sequencing problem gets more important when a larger number of pickers, i.e. more possibilities of assigning orders, have to be taken into account.

Summing up, it can be stated that the application of two constructive approaches leads to significant improvements with respect to the total tardiness of all customer orders. On average, across all problem classes the total tardiness can be reduced by 14.2% without a noticeable increase of the computing time required for generating an initial solution.

6.2.2 Impact of the neighborhood structures

When designing a VND, another very important issue refers to the choice of the neighborhood structures and the sequence according to which they are used within the algorithm. The VND approach proposed by Henn (2015) performed quite well for the JOBSP. The author suggested utilization of the batch-related neighborhood structure $\mathcal{N}_1$ and the order-related structures $\mathcal{N}_3$ to $\mathcal{N}_6$. We also use these structures and the corresponding sequence. Furthermore, we introduce a new neighborhood structure $\mathcal{N}_2$ in order to be able to break up a complete batch and reinsert the orders into other batches. The impact of the moves applied according to the neighborhood structures is depicted in Table 3.

Table 3: Impact of the neighborhood structures for the case of five pickers and 200 orders

C	γ	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$
45	0.6	25.9	37.8	6.0	11.6	0.9	17.8
45	0.7	42.9	5.8	28.3	8.7	6.3	8.1
45	0.8	33.4	2.1	7.0	9.5	35.1	12.9
75	0.6	3.9	76.0	4.1	6.4	0.2	9.3
75	0.7	12.9	36.8	8.6	18.2	0.9	22.5
75	0.8	26.2	11.1	22.4	12.9	14.8	12.6
average		24.2	28.3	12.7	11.2	9.7	13.9

For problem classes with 5 order pickers and 200 customer orders, Table 3 includes information about the average proportion [in %] of the improvement obtained within the local search phases according to the neighborhood structure $\mathcal{N}_l$ ($l \in \{1, \ldots, 6\}$). For example, the entry 25.9 ($C = 45$, $\gamma = 0.6$, $l = 1$) means that 25.9% of the total improvement obtained by the VND can be attributed to the local search phases which run in the first neighborhood structure $\mathcal{N}_1$. The average proportion of the improvement ranges from 9.7% ($\mathcal{N}_5$) to 28.3% ($\mathcal{N}_2$). Therefore, it can be concluded that all neighborhood structures should be integrated into the algorithm.

The batch-related neighborhood structure $\mathcal{N}_1$ is responsible for 24.2% of the total improvement. It can be observed that the impact of this structure is much higher than the impact of the order-related neighborhood structures $\mathcal{N}_3$ to $\mathcal{N}_6$, whose proportion of the improvement amounts to approximately 10%, respectively. This can be traced back to the fact that the batch-related neighborhood structure is the first structure of the sequence and applied much more often than the order-related structures. Nevertheless, the impact of structures $\mathcal{N}_3$ to $\mathcal{N}_6$ should not be underestimated as improvements found in the corresponding local search phases may allow the algorithm to further improve the solution by continuing with $\mathcal{N}_1$ and $\mathcal{N}_2$.

The newly proposed neighborhood structure $\mathcal{N}_2$ shows the largest proportion (28.3%) of the total improvement, although it is sequenced after $\mathcal{N}_1$. The proportion strongly fluctuates ranging from 2.1% ($C = 45$, $\gamma = 0.8$) to 76.0% ($C = 75$, $\gamma = 0.6$). The reason can be found in the generation of the initial solution. As shown in the previous subsection, the ESDR-based algorithm leads to good solutions for loose due dates which are easy to meet (instances with a low MTCR γ or a medium MTCR and a large capacity C), while it is outperformed by the seed algorithm when the due dates get tight. A

move according to $\mathcal{N}_2$ is defined by breaking up a complete batch and reassigning the orders to other batches. If the initial solution is constructed by means of the seed algorithm, these moves will usually not lead to improvements since the orders are already grouped into batches in a reasonable manner which also takes into account the processing times. This is not true for solutions generated by applying the ESDR-based algorithm. In this approach, batches are constructed considering the due dates of the orders only, which is the reason why the impact of $\mathcal{N}_1$ (which simply changes the position of batches) is quite small. However, by means of moves according to $\mathcal{N}_2$, the batching can be optimized regarding the processing times resulting in massive improvements with respect to the total tardiness of all customer orders.

6.2.3 VND: Considerations regarding the solution quality and the computing time

In order to evaluate the performance of the VND approach, the algorithm is compared to two other solution approaches with respect to the total tardiness obtained. As a first benchmark, solutions provided by the ESDR-based algorithm in combination with the S-shape routing strategy are considered. The ESDR-based algorithm is a very straightforward approach to solve the JOBASRP and has also been used by Henn (2015). Since the solution quality of both the ESDR-based algorithm and the S-shape strategy is strongly dependent on the characteristics of the problem instance, we also use the initial solution of the VND in combination with the LKH heuristic as benchmark. The corresponding results from the experiments are presented in Tables 4 and 5.

In Tables 4 and 5, the average total tardiness (tar) in minutes is depicted for the ESDR-based algorithm combined with S-shape routing (ESDR), the initial solution after the application of the LKH heuristic (Initial) and the complete variable neighborhood descent algorithm (VND) for problem classes with 100 and 200 orders, respectively. Furthermore, the average improvements [in %] are presented compared to the solution provided by the ESDR-based algorithm (imp_1) and the initial solution of the VND (imp_2).

When comparing solutions obtained by the ESDR-based algorithm with VND solutions, significant improvements regarding the total tardiness of all customers can be observed. On average, the reduction of the total tardiness ranges from 51.5% (200 orders, 2 pickers, $C = 75$, $\gamma = 0.6$, $q = 1$) to 95.2% (200 orders, 3 pickers, $C = 45$, $\gamma = 0.7$, $q = 2$). The number of pickers does not seem to have a strong impact on the amount of improvement as the average reduction is between 69.2% and 71.2% for 100 customer orders and ranges from 73.5% to 74.8% for 200 orders. However, the impact of the other factors under investigation, namely the number of orders, the capacity C of the picking device, the MTCR γ as well as the number of blocks q does not seem to be negligible.

The more orders have to be assigned to batches, the larger the number of solutions gets, since more combinations of orders exist which can be grouped into a batch. In the ESDR-based algorithm, orders are sequentially assigned to batches not taking advantage of the larger number of possibilities. In contrast to this, the VND considers much more moves according to the neighborhood structures $\mathcal{N}_3$ to $\mathcal{N}_6$. Thus, it is not surprising that the amount of improvement increases with an increasing number of customer orders.

Table 4: Evaluation of the VND for 100 orders

			2 pickers							3 pickers							5 pickers						
C	γ	q	**ESDR**	**Initial**		**VND**				**ESDR**	**Initial**		**VND**				**ESDR**	**Initial**		**VND**			
			tar	tar	imp_1	tar	imp_1	imp_2	time	tar	tar	imp_1	tar	imp_1	imp_2	time	tar	tar	imp_1	tar	imp_1	imp_2	time
45	0.6	1	181	98	45.6	54	70.1	45.1	77	153	86	43.6	46	69.8	46.4	80	121	73	39.2	43	64.6	41.9	96
45	0.6	2	180	55	69.3	36	80.0	34.7	52	340	122	64.1	55	83.7	54.6	70	281	100	64.5	54	80.9	46.3	68
45	0.6	3	525	107	79.5	65	87.5	39.1	48	716	156	78.2	87	87.8	44.2	72	1052	236	77.6	148	85.9	37.2	88
45	0.7	1	2754	840	69.5	359	87.0	57.3	293	2061	906	56.0	312	84.9	65.6	377	1621	824	49.2	323	80.0	60.8	416
45	0.7	2	4647	1140	75.5	545	88.3	52.2	198	3712	1452	60.9	665	82.1	54.2	264	2415	1099	54.5	451	81.3	58.9	289
45	0.7	3	7751	1606	79.3	995	87.2	38.1	160	5544	1605	71.0	741	86.6	53.8	252	3886	1535	60.5	749	80.7	51.2	268
45	0.8	1	9145	4614	49.6	3590	60.7	22.2	342	6880	3723	45.9	2826	58.9	24.1	371	5705	3107	45.5	2362	58.6	24.0	259
45	0.8	2	12773	6076	52.4	4859	62.0	20.0	216	8730	4422	49.3	3345	61.7	24.3	260	7780	3716	52.2	2892	62.8	22.2	292
45	0.8	3	16987	7123	58.1	5750	66.1	19.3	204	11980	5363	55.2	4306	64.1	19.7	247	4301	2526	41.3	1873	56.4	25.8	367
75	0.6	1	71	57	19.1	33	53.2	42.1	122	96	77	19.1	35	63.0	54.2	126	130	107	17.8	35	73.3	67.5	151
75	0.6	2	81	58	28.2	30	62.5	47.8	46	122	90	26.0	42	65.4	53.3	49	167	118	29.6	46	72.5	60.9	57
75	0.6	3	119	65	45.1	41	65.6	37.4	50	203	100	50.6	53	74.0	47.4	59	350	193	44.9	109	68.9	43.6	72
75	0.7	1	174	127	27.4	83	52.4	34.5	184	211	142	32.9	84	60.3	40.8	164	319	235	26.2	143	55.3	39.4	243
75	0.7	2	289	153	47.2	105	63.8	31.3	72	528	279	47.3	171	67.7	38.6	104	462	270	41.6	161	65.1	40.2	100
75	0.7	3	750	205	72.7	154	79.5	24.9	85	798	268	66.4	187	76.5	30.1	103	1083	377	65.2	257	76.3	32.0	120
75	0.8	1	3397	2204	35.1	1170	65.6	46.9	545	2969	2206	25.7	1211	59.2	45.1	685	3046	2077	31.8	1200	60.6	42.3	310
75	0.8	2	5319	3066	42.4	1737	67.3	43.3	276	3961	2446	38.2	1315	66.8	46.2	319	4487	2558	43.0	1563	65.2	38.9	376
75	0.8	3	8238	3792	54.0	2294	72.2	39.5	338	6243	3333	46.6	1972	68.4	40.8	431	2193	1672	23.8	931	57.6	44.3	596
average			4077	1744	52.8	1217	70.6	37.5	184	3069	1488	48.7	970	71.2	43.5	224	2189	1157	44.9	741	69.2	43.2	232

Table 5: Evaluation of the VND for 200 orders

			2 pickers							3 pickers							5 pickers						
C	γ	q	**ESDR**	**Initial**		**VND**				**ESDR**	**Initial**		**VND**				**ESDR**	**Initial**		**VND**			
			tar	tar	imp_1	tar	imp_1	imp_2	time	tar	tar	imp_1	tar	imp_1	imp_2	time	tar	tar	imp_1	tar	imp_1	imp_2	time
45	0.6	1	228	114	49.9	63	72.6	45.2	222	266	141	46.8	63	76.3	55.4	349	253	143	43.6	76	69.8	46.5	448
45	0.6	2	295	99	66.5	53	82.2	46.7	164	284	91	67.7	58	79.5	36.4	229	591	203	65.7	98	83.5	51.8	501
45	0.6	3	1978	232	88.3	97	95.1	58.1	256	1008	135	86.6	94	90.7	30.5	252	1026	199	80.6	130	87.3	34.6	474
45	0.7	1	10219	2304	77.5	671	93.4	70.9	1494	7816	2959	62.1	552	92.9	81.3	2553	4745	2224	53.1	433	90.9	80.5	3091
45	0.7	2	16722	3038	81.8	1040	93.8	65.8	1130	11392	3475	69.5	551	95.2	84.2	2022	7071	2491	64.8	505	92.9	79.7	2680
45	0.7	3	28590	3980	86.1	1815	93.7	54.4	1348	19366	5192	73.2	1455	92.5	72.0	2597	13585	5052	62.8	1475	89.1	70.8	3347
45	0.8	1	37946	18075	52.4	14069	62.9	22.2	2263	25665	13208	48.5	9535	62.8	27.8	2990	16298	8966	45.0	6311	61.3	29.6	3621
45	0.8	2	50187	22667	54.8	17632	64.9	22.2	1951	32649	15722	51.8	11283	65.4	28.2	2264	21624	11260	47.9	7912	63.4	29.7	2875
45	0.8	3	66036	26109	60.5	20557	68.9	21.3	1918	45702	19245	57.9	14804	67.6	23.1	2488	28415	13058	54.0	9511	66.5	27.2	3155
75	0.6	1	78	63	19.1	37	51.9	40.5	231	98	78	20.6	37	62.7	53.0	255	156	123	21.1	54	65.1	55.8	334
75	0.6	2	90	63	29.8	33	62.9	47.1	101	123	89	28.0	43	65.2	51.6	121	213	148	30.7	71	66.8	52.1	184
75	0.6	3	139	63	54.6	36	73.7	42.0	106	204	114	43.8	68	66.7	40.7	141	340	196	42.2	103	69.7	47.6	218
75	0.7	1	229	146	36.3	99	56.6	31.9	406	242	163	32.5	93	61.7	43.3	449	316	236	25.2	137	56.7	42.1	518
75	0.7	2	560	326	41.7	218	61.1	33.3	185	361	193	46.5	126	65.0	34.6	209	437	246	43.8	135	69.0	44.9	260
75	0.7	3	924	221	76.1	160	82.7	27.7	209	1143	317	72.2	221	80.6	30.3	231	1414	393	72.2	264	81.3	32.7	388
75	0.8	1	13381	8172	38.9	3687	72.4	54.9	2472	9484	6704	29.3	2804	70.4	58.2	3075	6763	5092	24.7	2280	66.3	55.2	3044
75	0.8	2	20149	10792	46.4	5165	74.4	52.1	1629	13026	7953	38.9	3148	75.8	60.4	1820	9832	6375	35.2	2913	70.4	54.3	1940
75	0.8	3	29080	12385	57.4	6127	78.9	50.5	1775	21485	11051	48.6	5156	76.0	53.3	2027	14462	8279	42.8	3922	72.9	52.6	2284
average			15379	6047	56.6	3975	74.6	43.7	992	10573	4824	51.4	2783	74.8	48.0	1337	7086	3593	47.5	2018	73.5	49.3	1631

An increasing capacity of the picking device leads to a reduction with respect to the relative improvement. A larger capacity leads to tours containing more pick locations. Since the S-shape strategy is known to perform quite well if the number of pick locations is large in comparison to the number of subaisles (Roodbergen, 2001), the difference between the S-shape strategy and the LKH heuristic in terms of solution quality is quite small in these cases, resulting in less space for improvement. The reverse line of argumentation holds for the impact of an increase in the number of blocks as more blocks result in more subaisles, leading to a smaller number of pick locations per subaisle. Thus, the solution quality of the S-shape policy deteriorates and the amount of the improvement obtained by applying the VND (into which two more sophisticated routing algorithms are integrated) increases. The MTCR determines how tight the due dates are. A larger MTCR leads to tighter due dates which tend to increase the total tardiness of solutions to the corresponding problem instance. Apart from problem classes with an MTCR $\gamma = 0.8$ and a capacity $C = 45$, the amount of improvement increases with an increasing MTCR. This can be explained by the fact that the solution quality of the ESDR-based algorithm deteriorates when the due dates get tighter (see Section 6.2.1). If the MTCR is low or medium ($\gamma \in \{0.6, 0.7\}$), the VND leads to solutions with an average total tardiness of up to 1815 minutes (200 orders, 2 pickers, $C = 45$, $\gamma = 0.7$, $q = 3$), i.e. the average tardiness of an order amounts to 9 minutes. For instances from these problem classes, the VND provides many solutions with a total tardiness which is zero or close to zero, resulting in improvements of 100% (regardless of the absolute amount of improvement). Thus, conclusions have to be drawn carefully for instances with a small MTCR. If the MTCR is large ($\gamma = 0.8$), the average total tardiness significantly increases. Finding solutions with a low total tardiness gets even harder when the capacity is small ($C = 45$) since the total processing times of the orders increases. Therefore, the amount of the relative improvement decreases when very difficult instances are considered. Nevertheless, it can be seen that the VND manages to massively reduce the total tardiness as an absolute improvement of up to 45479 minutes (200 orders, 2 pickers, $C = 45$, $\gamma = 0.8$, $q = 3$) is achieved, which corresponds to a decrease of the tardiness by approximately 4 hours per order.

In order to investigate the performance of the VND, we also compare the solutions to the initial solution in combination with the LKH heuristic which represents a much stronger benchmark than the ESDR-based algorithm combined with the simple S-shape strategy. It can be observed that the VND manages to reduce the total tardiness by between 19.3% (100 orders, 2 pickers, $C = 45$, $\gamma = 0.8$, $q = 3$) and 84.2% (200 orders, 3 pickers, $C = 45$, $\gamma = 0.7$, $q = 2$). On average, across all problem classes, the improvement amounts to 44.2%, which demonstrates that the application of the VND is pivotal for the generation of high-quality solutions. Three main factors can be identified which have an impact on the amount of the improvement obtained. First, as observed in the comparison to the ESDR-based algorithm, a larger number of orders allows for more space of improvement. Second, the amount of improvement tends to decrease with an increasing number of blocks. Third, the combination of the MTCR γ and the capacity C has a great impact. While the largest improvements can be obtained for $\gamma = 0.7$ and $C = 45$, the smallest reductions are observed for $\gamma = 0.8$ and $C = 45$. This can be explained by the way how the initial solution is generated. The combination $\gamma = 0.8$ and $C = 45$ results in instances with very tight due dates which are very difficult to meet. In this case, the seed algorithm leads to very

good solutions and, therefore, is chosen for the construction of initial solutions. Due to the high quality of these solutions, only small improvements can be obtained by application of the VND. In contrast, the combination $\gamma = 0.7$ and $C = 45$ leads to due dates not tight enough for the seed algorithm but too tight for the ESDR-based algorithm. Thus, the quality of the initial solution is quite poor and the VND manages to significantly reduce the total tardiness.

Apart from the solution quality, the VND is evaluated with respect to the computing times required. Henn (2015) proposed solution approaches to the JOBSP using very simple routing strategies and reported computing times of up to 25 minutes for problem instances with 200 orders. It can be expected that our solution approach requires a higher computational effort as we integrated more complex routing algorithms. Furthermore, Henn (2015) considered a single-block layout only. Especially when the capacity is large and the simple S-shape strategy is used, many batches will result in the same tours as all picking aisles will be traversed. Due to this characteristic, the problem is reduced to a sequencing problem and their solution approaches will terminate much faster as there is less room for improvement. This is not true for our setup since we also consider more complex layouts. Furthermore, tours constructed by the LKH heuristic are dependent on the certain pick locations instead of the picking aisles to be visited only as it is the case for the S-shape strategy.

In our numerical experiments, it can be seen that the computing times are dependent on the number of customer orders as well as on how difficult the due dates are to be met. Of course, a larger number of orders significantly increases the computing times as it allows for much more possible moves regarding the neighborhood structures. For problem classes with 100 orders, the computing times are not a critical issue as they range from 1 to 11 minutes. When 200 orders are considered, however, up to 1 hour is required for solving a single instance. The largest computing times can be observed for problems with a large MTCR ($\gamma = 0.8$) and a small capacity ($C = 45$), which can be explained by the fact that these instances are characterized by very tight due dates. In practical applications, such high computing times may be a critical issue, which is why we investigated to which extent the solution quality decreases if less computing time is spent. Since the highest computing times arise in instances with 200 orders, 5 pickers, $C = 45$, $\gamma = 0.8$ and $q = 1$, we depicted the development of the solution quality over time for these instances in Fig. 5.

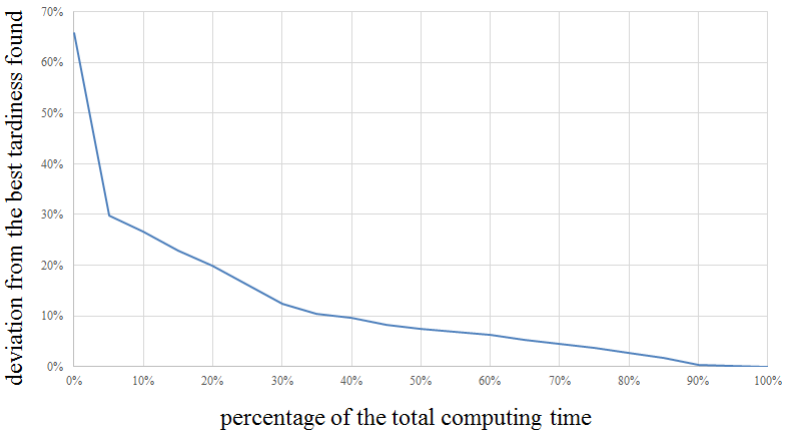

Fig. 5: Development of the solution quality over time

The VND starts with an initial solution in which the combined heuristic is used in order to determine the processing times of the batches. The computing time for constructing this solution is negligible. Then, the LKH heuristic is applied, significantly reducing the processing times and improving the objective function value. On average, application of the LKH heuristic to all batches only requires 12 seconds of computing time for instances from this problem class. The remaining improvement is obtained by the moves corresponding to the different neighborhood structures. As can be seen in Fig. 5, the largest proportion of the improvement is realized within the first 30% of the computing time. After investing 40% of the time, which corresponds to 24 minutes, the total tardiness is 10% above the tardiness of the best solution found after the VND terminates. However, it can be observed that the VND manages to steadily improve the solution until the end of the solution process, i.e. reducing the computing time will definitely result in a larger total tardiness.

If computing times are a critical issue and the solution process has to be speeded up, the removal of $\mathcal{N}_6$ would be a straightforward way as this neighborhood structure requires the highest computational effort. Considering instances from the problem class defined above, a removal of $\mathcal{N}_6$ would save 44% of the total computing time, while the average total tardiness would increase by 6.4%. Thus, we conclude that the VND can be adjusted, if necessary, in such a way that solutions with a reasonable quality can be found within an acceptable amount of computing time.

7 Conclusion

In this article, we considered the Joint Order Batching, Sequencing and Routing Problem which is rarely addressed in the literature, although it is pivotal for an efficient organization of manual order picking systems. We proposed a new mathematical model formulation. In contrast to existing formulations, the size of the model increases polynomially with the number of customer orders, which allows for solving small instances to optimality within a reasonable amount of computing time. Furthermore, we designed a variable neighborhood descent approach able to deal with very large problems. In order to reach the local optimum quite fast and speed up the solution process, a new heuristic for the construction of an initial solution is developed which outperforms the earliest start date rule-based algorithm by far when the due dates are very tight.

By means of numerical experiments, the solution quality of the variable neighborhood descent algorithm is evaluated. First, the initial solution is compared to solutions obtained by applying the earliest start date rule-based algorithm, which represents a common approach to generate a solution and was also used by Henn (2015). It is demonstrated that our initial solution leads to a reduction of the total tardiness by up to 63%. In a second step, we show that all neighborhood structures are important in order to obtain high quality solutions. The largest proportion of the total improvement is achieved by moves regarding a newly designed neighborhood structure which breaks up complete batches and reassigns the orders to other batches. Finally, the variable neighborhood descent algorithm is compared to two constructive approaches with respect to the solution quality. Combining the earliest start date rule-based algorithm

with the simple S-shape strategy represents a very straightforward way to solve the problem. However, it is pointed out that the solution quality of this approach is very poor as the variable neighborhood descent manages to improve the objective function value of these solutions by up to 95%, which means a massive reduction of the total tardiness.

We dealt with the joint batching, sequencing and routing problem in warehouses with wide aisles which allow order pickers for overtaking each other. Further research could concentrate on picker blocking aspects arising in narrow-aisle warehouses, where subaisles may be blocked when two pickers are moving in opposite directions or traffic jams may occur when several pickers have to visit the same storage location at the same time. When taking blocking aspects into account, tours of different pickers cannot be independently determined anymore, making the problem much more difficult to deal with. From a practical point of view, the online variant of the joint batching, sequencing and routing problem would also be a very interesting topic as customer orders arrive during the day and are not known in advance, which requires for very fast solution approaches in order to be able to recalculate each time a certain number of new orders has arrived.

References

Bozer, Y. A. & Kile, J. W. (2008): Order Batching in Walk-and-Pick Order Picking Systems. *International Journal of Production Research* 46, 1887-1909.

Burkard, R.; Deĭneko, V. G; van der Veen, J. A. A. & Woeginger, G. J. (1998): Well-Solvable Special Cases of the Traveling Salesman Problem: A Survey. *SIAM Review* 40, 496-546.

Chen, T.-L.; Cheng, C.-Y.; Chen, Y.-Y. & Chan, L.-K. (2015): An Efficient Hybrid Algorithm for Integrated Order Batching, Sequencing and Routing Problem. *International Journal of Production Economics* 159, 158-167.

Clarke, G. & Wright, J. W. (1964): Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research* 12, 568-581.

de Koster, R.; Le-Duc, T. & Roodbergen, K. J. (2007): Design and Control of Warehouse Order Picking: A Literature Review. *Science Direct* 182, 481-501.

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

Elsayed, E.; Lee, M. K.; Kim, S. & Scherer, E. (1993): Sequencing and Batching Procedures for Minimizing Earliness and Tardiness Penalty for Order Retrievals. *International Journal of Production Research* 31, 727-738.

Elsayed, E. & Lee, M. K. (1996): Order Processing in Automated Storage/Retrieval Systems with Due Dates. *IIE Transactions* 28, 567-577.

Gademann, N.; van den Berg, J. & van der Hoff, H. (2001): An Order Batching Algorithm for Wave Picking in a Parallel-Aisle Warehouse. *IIE Transactions* 33, 385-398.

Gademann, N. & van de Velde, S. (2005): Order Batching to Minimize Total Travel Time. *IIE Transactions* 37, 63-75.

Gibson, D. R. & Sharp, G. P. (1992): Order Batching Procedures. *European Journal of Operational Research* 58, 57-67.

Grosse, E. H.; Glock, C. H. & Ballester-Ripoll, R. (2014): A Simulated Annealing Approach for the Joint Order Batching and Order Picker Routing Problem with Weight Restrictions. *International Journal of Operations and Quantitative Management* 20, 65-83.

Gu, J.; Goetschalckx, M. & McGinnis, L. F. (2007): Research on Warehouse Operation: A Comprehensive Review. *European Journal of Operational Research* 177, 1-21.

Hall, R. W. (1993): Distance Approximations for Routing Manual Pickers in a Warehouse. *IIE Transactions* 25, 76-87.

Hansen, P. & Mladenović, N. (2001): Variable neighborhood search: principles and applications. *European Journal of Operational Research* 130, 449-467.

Helsgaun, K. (2000): An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic. *European Journal of Operational Research* 126, 106-130.

Henn, S.; Koch, S. & Wäscher, G. (2012): Order Batching in Order Picking Warehouses: A Survey of Solution Approaches. *Warehousing in the Global Supply Chain: Advanced Models, Tools and Applications for Storage Systems*, Manzini, R. (eds.), 105-137, Springer: London.

Henn, S. & Schmid, V. (2013): Metaheuristics for Order Batching and Sequencing in Manual Order Picking Systems. *Computers & Industrial Engineering* 66, 338-351.

Henn, S. (2015): Order Batching and Sequencing for the Minimization of the Total Tardiness in Picker-to-Part Warehouses. *Flexible Services and Manufacturing* 27, 86-114.

Hong. S.; Johnson, A. L. & Peters B. A. (2012): Batch Picking in Narrow-Aisle Order Picking Systems with Consideration for Picker Blocking. *European Journal of Operational Research* 221, 557-570.

Jarvis, J. M. & McDowell, E. D. (1991): Optimal Product Layout in an Order Picking Warehouse. *IIE Transactions* 23, 93-102.

Koulamas, C. (1994): The Total Tardiness Problem: Review and Extensions. *Operations Research* 42, 1025-1041.

Kulak, O.; Sahin, Y. & Taner, M. E. (2012): Joint Order Batching and Picker Routing in Single and Multiple-Cross-Aisle Warehouses Using Cluster-Based Tabu Search Algorithms. *Flexible Services and Manufacturing Journal* 24, 52-80.

Letchford, A. N.; Nasiri, S. D. & Theis, D. O. (2013): Compact Formulations of the Steiner Traveling Salesman Problem and Related Problems. *European Journal of Operational Research* 228, 83-92.

Muter, I. & Öncan, T. (2015): An Exact Solution Approach for the Order Batching Problem. *IIE Transaction* 47, 728-738.

Öncan, T. (2015): MILP Formulations and an Iterated Local Search Algorithm with Tabu Thresholding for the Order Batching Problem. *European Journal of Operational Research* 243, 142-155.

Petersen, C. G. & Schmenner, R. W. (1999): An Evaluation of Routing and Volume-Based Storage Policies in an Order Picking Operation. *Decision Science* 30, 481-501.

Pinedo, M. L. (2016): Scheduling: Theory, Algorithms, and Systems. 5th edition, Springer, Cham et al.

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

Roodbergen, K. J. (2001): Layout and Routing Methods for Warehouses. Trial: Rotterdam.

Roodbergen, K. J. & de Koster, R. (2001a): Routing Order Pickers in a Warehouse with a Middle Aisle. *European Journal of Operational Research* 133, 32-43.

Roodbergen, K. J. & de Koster, R. (2001b): Routing Methods for Warehouses with Multiple Cross Aisles. *International Journal of Production Research* 39, 1865-1883.

Scholz, A. & Wäscher, G. (2015): A Solution Approach for the Joint Order Batching and Picker Routing Problem in a Two-Block Layout. Working Paper No. 4/2015, Faculty of Economics and Management, Otto-von-Guericke University Magdeburg.

Scholz, A.; Henn, S.; Stuhlmann, M. & Wäscher, G. (2016): A New Mathematical Programming Formulation for the Single-Picker Routing Problem. *European Journal of Operational Research* 253, 68-84.

Theys, C.; Bräysy, O.; Dullaert, W. & Raa, B. (2010): Using a TSP Heuristic for Routing Order Pickers in Warehouses. *European Journal of Operational Research* 200, 755-763.

Tompkins, J. A.; White, J. A.; Bozer, Y. A. & Tanchoco, J. M. A. (2010): Facilities Planning. 4th edition, John Wiley & Sons, New Jersey.

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

Wäscher, G. (2004): Order Picking: A Survey of Planning Problems and Methods. *Supply Chain Management and Reverse Logistics*, Dyckhoff, H.; Lackes, R. & Reese, J. (eds.), 324-370, Springer: Berlin.

Otto von Guericke University Magdeburg
Faculty of Economics and Management
P.O. Box 4120 | 39016 Magdeburg | Germany

Tel.: +49 (0) 3 91 / 67-1 85 84
Fax: +49 (0) 3 91 / 67-1 21 20

www.fww.ovgu.de/femm

ISSN 1615-4274