WORKING PAPER SERIES

A Solution Approach for the Joint Order Batching and Picker Routing Problem in a Two-Block Layout

André Scholz/Gerhard Wäscher

Working Paper No. 4/2015

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 and 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

A Solution Approach for the Joint Order Batching and Picker Routing Problem in a Two-Block Layout

A. Scholz, G. Wäscher

Abstract

Order Batching and Picker Routing Problems arise in warehouses when articles have to be retrieved from their storage location in order to satisfy a given demand specified by customer orders. The Order Batching Problem includes the grouping of a given set of customer orders into feasible picking orders such that the total length of all picker tours is minimized. The problem of determining the sequence according to which articles have to be picked is known as the Picker Routing Problem. Although these problems occur at the same planning level, it is common to solve these problems not simultaneously, but separately and in sequence. As for the batching problem it is usually assumed that the order pickers, when making their ways through the warehouse, follow a certain, simple routing strategy. Based on this routing strategy, the customer orders are grouped into picking orders. The advantage of this approach can be seen in the fact that – in particular for single-block warehouse layouts – the corresponding order picker tours are very straightforward and can be memorized easily by the order pickers. This advantage diminishes, however, when more complex, multi-block layouts have to be dealt with. Furthermore, in such case, the approach may result in picker tours that can be far from optimal. Therefore, for multi-block layouts, we develop a new approach, namely an iterated local search algorithm into which different routing algorithms have been integrated and which allows for solving the Order Batching Problem and the Picker Routing Problem simultaneously. By means of numerical experiments it is shown that this approach results in a substantial improvement of the solution quality without increasing computing times.

Keywords: Order Picking, Order Batching, Picker Routing, Iterated Local Search

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 warehouse function dealing with the retrieval of articles from their storage location in order to satisfy a given demand specified by customer orders (Petersen & Schmenner, 1999; Wäscher, 2004). It occurs because incoming articles are received and stored in large volume unit loads while customers order small volumes (less-than-unit-loads) of different articles. Even though there have been different attempts to automate the picking process, approximately 80% of all order picking systems in Western Europe are manual ones (de Koster et al., 2007). Among such systems, picker-to-parts systems can be looked upon as the most important ones, where order pickers walk (or drive) through the warehouse and collect the requested articles from the different storage locations (Wäscher, 2004). Due to a large-scale involvement of human operators, order picking includes the most cost-intensive warehouse operations. According to the literature, between 50% (Frazelle, 2002) and 65% (Coyle et al., 1996) of the total warehouse operating costs can be attributed to order picking.

The provision of good-quality solutions to the decision problems on the operative planning level can be considered as a key to an efficient organization of order picking operations (de Koster et al., 1999a). In particular, two problems can be identified which have to be dealt with thoroughly, namely the Order Batching Problem and the Picker Routing Problem. In the Order Batching Problem, a set of (indivisible) customer orders is given, each of which requiring certain articles to be collected from known storage locations within the warehouse. Based on – at least basic – assumptions about how the persons who collect the articles (order pickers) will proceed from one location to the next, the customer orders have to be grouped into picking orders (batches) such that the total length of all picking tours necessary to collect all articles is minimized. In the Picker Routing Problem, a set of picking orders is given, and, for each picking order, one has to determine a tour of minimum length that allows for collecting all articles of the respective picking order. Obviously, both problems are closely interconnected. Solving the batching problem requires information on the sequence according to which the locations of the articles are visited by the order pickers, while solving the routing problems requires information on how the customer orders are grouped into picking orders.

In practice, these two problems are usually solved successively. It is assumed that the order pickers, when making their way through the warehouse, follow a certain routing strategy. Based on this routing strategy, the customer orders are grouped into picking orders. Despite the fact that – at least for basic (single-block and two-block) warehouse layouts (Ratliff & Rosenthal, 1983; Roodbergen & de Koster, 2001a) – efficient exact algorithms exist for solving the routing problem, such an optimal routing strategy is hardly ever applied. Instead, the Order Batching Problem is solved under the assumption that the order pickers follow a strategy in which the locations to be visited are sequenced heuristically. The most frequently used (heuristic) routing strategies are s-shape and largest gap routing.

In order to justify the application of such heuristic strategies, it is argued that the provided solutions (tours) appear to be “more straightforward”; they can probably be memorized more easily than optimal ones and, therefore, get more easily accepted by the order pickers (de Koster et al., 1999a). This reasoning is questionable, though. Firstly, the solution quality of heuristic routing strategies is strongly dependent

on the problem data (e.g. number of picking aisles of the warehouse, capacity of the picking device), and it is not uncommon that they provide tours which are far from being optimal. Secondly, for more complex layouts (e.g. layouts with two or more blocks), even these routing strategies may result in picking tours which are not straightforward at all but equally complex as optimal tours (see Roodbergen & de Koster (2001b) for example tours). In other words, the core argument in favor of heuristic routing strategies, the simple structure of the provided tours, is no longer valid. Thirdly, also the quality of the solutions generated by means of the s-shape or largest gap routing strategies tends to deteriorate when more complex warehouse layouts are considered (Roodbergen, 2001).

We conclude that, for efficient order picking in more complex, multi-block layouts, solving the Picker Routing Problem must not be separated from solving the Order Batching Problem. Hence, we propose an approach which allows for solving the Joint Order Batching and Picker Routing Problem and into which algorithms for order batching and picker routing are integrated. In order to keep the exposition simple, the approach will be described and exemplified for two-block layouts, only. Nevertheless, it will become clear that the approach can be extended to block layouts of higher dimensions and that the observed benefits hold or even increase for such layouts.

We further note that, on the one hand, more sophisticated routing algorithms will result in shorter picker tours and, consequently, in better solutions for the joint batching and routing problem. On the other hand, however, the integration of complex routing algorithms may result in longer, probably unacceptable computing times. We will, therefore, also study under which conditions exact routing algorithms can be applied, and we will suggest heuristic modifications of an exact routing algorithm for situations in which this is not possible.

The remainder of this paper is organized as follows: In section 2, we give a precise statement of the Joint Order Batching and Picker Routing Problem. Section 3 comprises a literature review regarding the routing, the batching, and the joint problem. For the joint problem, a corresponding mathematical model formulation based on Gademann & van de Velde (2005) is presented in section 4. Section 5 contains solution approaches for the Picker Routing Problem, namely the exact algorithm by Roodbergen & de Koster (2001a), the s-shape, largest gap, aisle-by-aisle and combined$^+$ heuristic. Furthermore, a heuristic solution approach derived from the exact algorithm is presented. In section 6, the iterated local search algorithm is introduced which we propose for the joint batching and routing problem. Section 7 is devoted to the numerical experiments which have been carried out with the proposed approach. We explain how the test problem instances were generated, and we report and discuss the results from the experiments. The paper concludes with an outlook on further research.

2 Problem description

In the following, we consider a distribution warehouse with a manual, low-level picker-to-part order picking system from which a given set of items has to be retrieved. In a manual order picking system, human operators perform the necessary tasks. In picker-to-parts systems the order pickers walk through

the picking area, visit the storage locations of the respective articles, and remove (pick) the required number of article units/items. In low-level picker-to-parts systems the items have to be removed from pallets or bins placed on the floor or from low-level racks which are directly accessible to the order pickers (Henn et al., 2012). The picking area possesses a block layout, i.e. it consists of a certain number of straight parallel picking aisles of equal length and width. The storage locations are of identical size and arranged on both sides of the picking aisles. By means of cross aisles, the order pickers are enabled to enter and exit a picking aisle. The part of the warehouse between two adjacent cross aisles is called a block and the corresponding part of the picking aisle is denoted as a subaisle, i.e. each picking aisle is composed of q subaisles, where q denotes the number of blocks. The order pickers enter the picking area at the depot. This is also the place where they return to in order to deposit the picked items. An example of a block layout with two blocks is depicted in Fig. 1. Warehouses with two blocks contain three cross aisles, namely the front, middle and rear cross aisle, where the front cross aisle presents the cross aisle nearest to the depot.

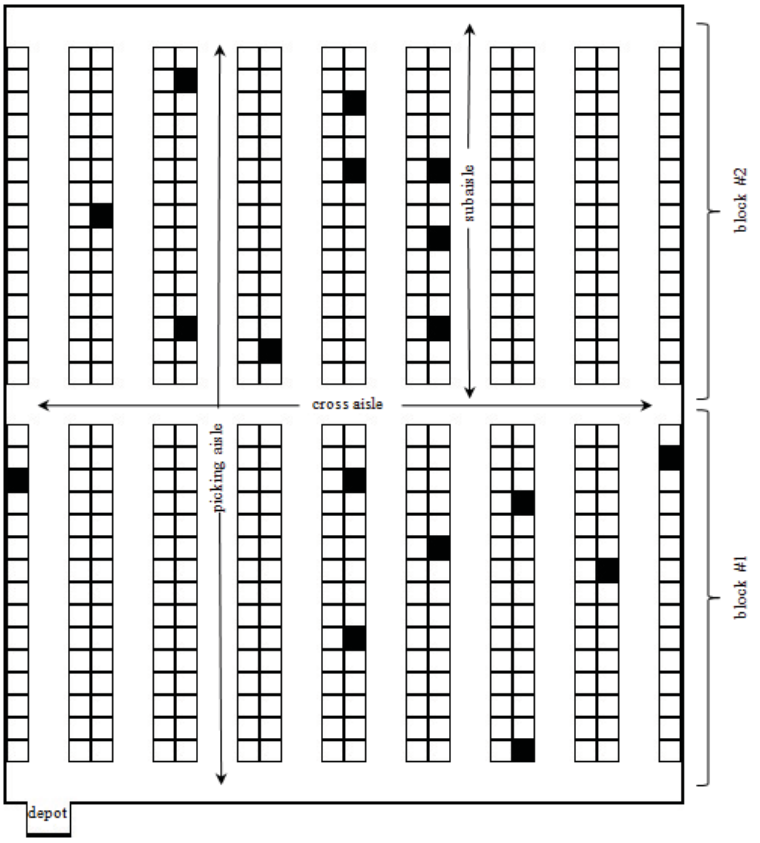


Fig. 1: Two-block layout

When moving through the warehouse, the order pickers use a picking device (e.g. a cart or a roll cage) on which they can place several items. Thus, they may visit several article locations before they return to the depot. In other words: the requested items are picked on tours through the warehouse. The number of stops on each tour is limited by the available space of the picking device on the one hand and by the capacity requirements of the items to be picked on the other hand. The information about the items to be retrieved from the warehouse is comprised in a set of customer orders. Each of these orders is related to a particular (external or internal) customer and consists of a set of order lines, each one identifying a

particular article and the corresponding quantity of this article. All items requested in the customer orders should be retrieved in a way such that the total length of the necessary picker tours (total tour length) is minimized. On their tours through the warehouse, order pickers are guided by pick lists. A pick list contains the order lines which should be processed together (picking order) and the storage locations of the respective articles. Furthermore, the order lines are already sorted into the sequence according to which the order picker is meant to visit the storage locations of the requested items. A picking order may be composed of a combination of multiple customer orders. Splitting of customer orders, however, is not permitted since it would result in an additional, unacceptable consolidation effort. In this case a picking order is addressed as a batch.

Let a (non-empty) set of customer orders be given, each of which requiring certain articles with known storage locations to be retrieved from a distribution warehouse. Then, in order to minimize the total length of the picker tours necessary to collect all requested articles, the following two questions have to be answered:

- How should the given set of customer orders be grouped (batched) into picking orders? (Order Batching Problem)
- For each picking order, in which sequence should the storage locations of the articles be visited which are included in the respective order? (Picker Routing Problem)

Obviously, both problems are closely interconnected and, therefore, should be solved simultaneously, giving rise to the Joint Order Batching and Picker Routing Problem (JOBPRP). Gademann & van de Velde (2005) have proven the NP-hardness of a variant of the Order Batching Problem in which a single-block layout is assumed and the capacity of the picking device is measured in the number of customer orders. Based on this conclusion, it is straightforward to show that the JOBPRP considered in this paper is also NP-hard. Before we provide a model formulation of the JOBPRP, we will review the state-of-the-art related to the individual problems and to the joint problem.

3 Literature review

3.1 Picker Routing Problem

Given a set of items to be collected and the respective locations where they are stored in the warehouse, a sequence must be determined according to which these locations should be visited such that the length of the corresponding picker tour is minimized. This so-called Picker Routing Problem (PRP) represents a special case of the classic Traveling Salesman Problem (TSP). For single-block layouts the PRP can be solved efficiently to optimality. Ratliff & Rosenthal (1983) present an exact algorithm with complexity $O(n+m)$ where n stands for the number of articles to be picked and m for the number of picking aisles. In practice, though, instead of proceeding according to optimal routes, which are often looked upon as complicated and difficult to memorize, order pickers seem to prefer routes based on simple routing strategies. The application of such routing strategies, e.g. the s-shape, the return, the midpoint and the

largest gap strategy, can be considered as a heuristic approach to the PRP. As for the s-shape heuristic (Goetschalckx & Ratliff, 1988), the order picker entirely traverses every picking aisle containing at least one item to be picked. When following the return heuristic, the order picker enters and leaves every picking aisle to be visited from the front cross aisle. For the midpoint heuristic, the warehouse is divided equally into two areas, the front section and the rear section. All articles located in the front section are accessed from the front cross aisle, while articles in the rear section are reached from the rear cross aisle. For the largest gap heuristic, in each aisle to be visited the two warehouse sections are determined by the largest distance between two adjacent requested articles or between a requested article and the adjacent cross aisle. Hall (1993) noticed that the largest gap heuristic outperforms the midpoint heuristic. Heuristics based on more sophisticated routing strategies are the composite heuristic (Petersen, 1997) and the aisle-by-aisle heuristic (Vaughan & Petersen, 1999) which combine elements of the s-shape and the return heuristic. These two heuristics have in common that, for each picking aisle where an article has to be picked, it has to be decided whether the order picker entirely traverses the picking aisle or, alternatively, enters and leaves it from the same cross aisle. For the aisle-by-aisle heuristic, this decision is made by dynamic programming. When using the composite heuristic a picking aisle is traversed if more than half of the picking aisle has to be entered to pick all items in this aisle.

All above-mentioned approaches to the PRP have been designed initially for single-block layouts. Roodbergen & de Koster (2001b) modified the Ratliff-Rosenthal Algorithm for the PRP in two-block layouts. For two-block and more complex layouts, the authors also introduced extensions of the s-shape and the largest gap heuristic. Furthermore, they proposed a combined heuristic, which includes elements of the aisle-by-aisle heuristic. Theys et al. (2010) applied different TSP heuristics to various classes of PRP instances related to warehouses with two and more blocks. They report significant (average) savings in the total tour length (up to 48% in comparison to tours constructed by the s-shape strategy) when using the Lin-Kernighan-Helsgaun Heuristic (Helsgaun, 2000).

3.2 Order Batching Problem

The Order Batching Problem (OBP) can be stated as follows: 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 sum of the total lengths of all tours required to collect the requested items is minimized? (Wäscher, 2004; Henn & Wäscher, 2012)

The OBP as described above is known to be NP-hard (in the strong sense) if the number of orders per batch is greater than two (Gademann & van de Velde, 2005). Hence, it is not surprising that only a few exact solution approaches have been proposed so far for this problem. Gademann & van de Velde (2005) presented a branch-and-price algorithm with column generation that was able to solve problem instances with up to 32 customer orders to optimality. Bozer & Kile (2008) proposed a mixed-integer programming approach that provided optimal solutions for instances with up to 25 orders within a few minutes. However, their approach is limited to s-shape routing and, thus, not suitable for the JOBPRP.

Apart from these exact approaches, a large variety of heuristic solution approaches exists. (For a very

detailed review, we refer to de Koster et al. (2007) and Henn et al. (2012).) The most prominent ones are priority rule-based algorithms, seed algorithms, savings algorithms and metaheuristics. In priority rule-based algorithms, priorities are assigned to customer orders at first; then, in the sequence given by the priorities, the customer orders are successively allocated to batches as long as the capacity of the picking device is not exceeded (Gibson & Sharp, 1992). Seed algorithms have been introduced by Elsayed (1981). Batches are generated sequentially in such algorithms. The procedure for each batch is as follows: In a first step, a seed (or initial) order is chosen from those orders not yet added to a batch. Then, in a second step, not yet selected customer orders are added to the seed order until no further order can be added without violating the capacity constraint of the picking device. Savings algorithms are based on the Clarke-and-Wright Algorithm for the Vehicle Routing Problem (Clarke & Wright, 1964) which has been adapted for the OBP. The initial version can be described as follows: At the beginning, savings (in terms of reductions of the total travel distance) are determined which can be obtained by collecting articles of two customer orders on a single (large) tour instead of collecting them on two separate tours. Then, in a non-ascending order of the savings, the pairs of customer orders are assigned to batches in a way in which one tries to include both orders in the same batch. The algorithm terminates when each order has been assigned to a batch or when all pairs of orders have been considered. A straightforward improvement of the algorithm consists of recalculating the savings each time a customer order is added to a batch (Elsayed & Unal, 1989). By means of extensive numerical studies, de Koster et al. (1999b) have shown that, among these constructive algorithms, either seed or savings algorithms provide the best solutions for the OBP.

Several metaheuristics have also been proposed for the OBP. Tsai et al. (2008) presented a genetic algorithm which is based on the assumption that splitting customer orders is allowed and, therefore, deals with a problem different to the one introduced above. Gademann & van de Velde (2005) suggested an iterated improvement algorithm in which, starting from an initial solution constructed by means of the first-come-first-served rule, improved solutions are obtained by so-called swap moves, i.e. by interchanging two customer orders from two different batches. They were able to solve instances with up to 30 customer orders within a few seconds. Albareda-Sambola et al. (2009) developed a variable neighborhood search algorithm for the OBP in which three different kinds of neighborhood structures are considered. This approach was applied to four different problem classes including up to 250 orders, and the authors reported computing times of less than one minute. Henn et al. (2010) proposed an iterated local search algorithm and a rank-based ant system. Comprehensive numerical studies have shown that these approaches lead to very good solutions but also require long computing times (up to 20 minutes) for large instances (up to 60 customer orders). Henn & Wäscher (2012) describe a tabu search algorithm and an attribute-based hill climber approach to the OBP which outperform the iterated local search algorithm in terms of solution quality and require computing times of up to two minutes (classic tabu search) and up to ten minutes (attribute-based hill climber) for problem instances with up to 100 customer orders, respectively.

3.3 Joint Order Batching and Picker Routing Problem

So far, only few articles exist dealing with the JOBPRP. Kulak et al. (2012) dealt with the problem in a single and a two-block block layout where the depot is located at the middle cross aisle instead of the front cross aisle. They provided a mixed-integer programming formulation based on formulations for the Bin Packing and the Traveling Salesman Problem. However, the number of subtour elimination constraints used in this model increases exponentially with the number of different pick locations. Therefore, a tabu search algorithm was used to solve the OBP. A nearest neighbor and a savings heuristic were applied in order to solve the arising routing problems.

Another approach to the joint problem in a single-block layout was presented by Grosse et al. (2014). However, they dealt with a case different to the one considered in this paper, namely a problem with a single customer order which may be split into different batches. They proposed a simulated annealing algorithm for the batching problem and combined it with four different routing algorithms. These routing algorithms were also used to create initial batches.

4 Model formulation

In order to formulate a model and develop a solution approach for the above-defined JOBPRP, we assume that each article has been assigned to exactly one storage location.

A straightforward way to formulate a model for the JOBPRP consists of considering – either explicitly or implicitly – all feasible batches and to choose some of these batches such that each customer order is contained in exactly one batch and the total tour length is minimized. A model formulation based on this approach has been introduced by Gademann & van de Velde (2005) and includes the following parameters:

The set J contains all customer orders and the set F all feasible batches. A batch is called feasible if it does not violate the capacity constraint resulting from the picking device. The constants a_{fj} indicate whether a customer order $j \in J$ is included in batch $f \in F$ ($a_{fj} = 1$) or not ($a_{fj} = 0$). Furthermore, the constants d_f represent the minimal length of an order picking tour which includes all articles of customer orders contained in batch $f \in F$. In order to calculate the constants d_f, for each feasible batch $f \in F$, the corresponding PRP has to be solved. Finally, the variables x_f indicate whether batch $f \in F$ is chosen ($x_f = 1$) or not ($x_f = 0$). Now the mathematical model for the JOBPRP can be formulated as follows:

$$\min \sum_{f \in F} d_f \cdot x_f \tag{1}$$

$$\sum_{f \in F} a_{fj} \cdot x_f = 1, \quad \forall j \in J; \tag{2}$$

$$x_f \in \{0, 1\}, \quad \forall f \in F. \tag{3}$$

Constraints (2) and (3) ensure that a set of batches is chosen in such a way that each customer order is contained in exactly one of these batches. The objective function represents the total tour length caused by the chosen batches.

In order to solve the above-presented mathematical model, it is necessary to consider (at least implicitly) all feasible batches. Furthermore, for each feasible batch the minimal length of the corresponding order picking tour has to be calculated. If a two-block layout of the warehouse is assumed, the PRP can be solved efficiently (Roodbergen & de Koster, 2001a). However, the number $|F|$ of feasible batches grows exponentially with the number $|J|$ of customer orders. Hence, only very small problems can be solved by applying the model formulation presented above.

For this reason, other approaches for solving the JOBPRP have to be considered. Before introducing the iterated local search algorithm for the solution of the joint problem, we present some heuristic routing strategies and the exact algorithm for the PRP in warehouses with two blocks provided by Roodbergen & de Koster (2001a).

5 Solution approaches for the PRP in two-block layouts

5.1 Exact solution approach

The PRP in warehouses with a two-block layout can be solved efficiently (Roodbergen & de Koster, 2001a). The corresponding exact solution approach will be presented briefly in the following. For a more detailed presentation of this algorithm we refer to Roodbergen & de Koster (2001a).

The solution approach consists of a dynamic programming procedure. The tour is constructed by starting with an empty tour and consecutively adding subaisles to the tour until all subaisles to be visited are included. The problem is represented as a graph $G = (V, E)$ with the set of edges E and the set of vertices $V = \bigcup_{j=1}^{m} \{a_j, b_j, c_j\} \cup \{v_0, \ldots, v_n\}$. v_0 represents the location of the depot and the vertex v_i represents the storage location of article $i \in \{1, \ldots, n\}$. Vertices a_j, b_j and c_j represent the points where picking aisle $j \in \{1, \ldots, m\}$ can be accessed, namely via the rear cross aisle, the middle cross aisle and the front cross aisle, respectively. Each pair of adjacent positions (either two storage locations of requested articles or a storage location and the adjacent cross aisle) is connected by two parallel edges. The corresponding edge weight is equal to the distance between these two positions. Fig. 2 depicts the graph-theoretical representation of the PRP introduced in Fig. 1.

From graph G edges have to be chosen such that they correspond to a feasible picking tour and the sum of the edge weights is minimized. A picking tour is called feasible if each vertex v_i ($i \in \{0, \ldots, n\}$) is included in the tour, each vertex has even or zero degree and, excluding vertices with zero degree, the subgraph is connected. Thus, an order picking tour can be perceived as a subgraph of G. Since an order picking tour is constructed by consecutively extending a subgraph of G, the following definitions are needed. Let L_j^- be a subgraph of G consisting of vertices a_j, b_j and c_j together with all vertices and (some) edges corresponding to the left of picking aisle j. Furthermore, L_j^{+1} denotes a subgraph of G

which additionally contains all vertices and (some) edges corresponding to subaisle j of block #1. L_j^{+2} denotes a subgraph that contains all vertices and (some) edges corresponding to the left of picking aisle $j+1$ except edges connecting aisle j and aisle $j+1$. The denotation L_j is used to indicate that a result holds for $L_j = L_j^-$, $L_j = L_j^{+1}$ or $L_j = L_j^{+2}$.

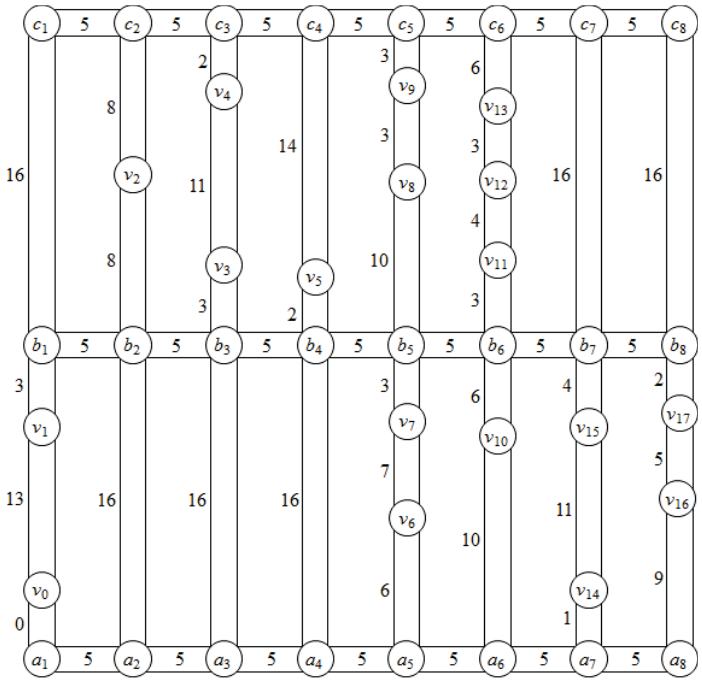


Fig. 2: Graph theoretical representation of a Picker Routing Problem

A subgraph T_j of L_j is called a L_j partial tour subgraph, if at least one subgraph of G (called completion) exists consisting of vertices and edges not included in L_j, such that the union of T_j and the completion leads to a graph representing an order picking tour. Hence, we can find an optimal order picking tour by considering a picking aisle j, constructing all L_j partial tour subgraphs (either L_j^-, L_j^{+1} or L_j^{+2} partial tour subgraphs) and looking for the best completion for each partial tour subgraph. In general, it is quite difficult to find the best completion, i.e. the completion with a minimum sum of edge weights, but we could consider L_m^{+2} partial tour subgraphs, where a graph with an empty set of edges would represent the best completion. However, the consideration of all partial tour subgraphs would lead to an unacceptable computational effort. Fortunately, not all partial tour subgraphs have to be taken into consideration since out of two L_j partial tour subgraphs with the same set of completions only the shorter subgraph, i.e. the subgraph with the smaller sum of edge weights, may lead to an optimal order picking tour. L_j partial tour subgraphs having an equal set of completions are called equivalent. Roodbergen & de Koster (2001b) have shown that 25 different equivalent classes exist which have to be taken into consideration. This theorem leads to the following solution approach:

The algorithm starts by constructing a L_1^- partial tour subgraph, i.e. a subgraph $T = \left(\tilde{V}, \tilde{E}\right)$ with $\tilde{V} = \{a_1, b_1, c_1\}$ and $\tilde{E} = \{\emptyset\}$. Then, the first subaisle of block #1 is considered and edges are added to T in such a way that each vertex in this subaisle is incident to at least two edges. Therefore, for the construction of an optimal order picking tour, only six different edge combinations have to be considered (Roodbergen & de Koster, 2001b). These edge combinations are depicted in Fig. 3 (a). Configurations (1)

and (6) indicate that a subaisle is traversed once or twice, respectively. The edge configurations (3) and (4) correspond to return strategies, i.e. the requested articles are picked by entering and leaving a subaisle via the same cross aisle, and configuration (5) follows a largest gap policy. Configuration (2) can only be used if a subaisle does not contain any requested articles. Otherwise, the inclusion of this configuration would not lead to a partial tour subgraph. Adding each of configurations (1) to (6) to a L_1^- partial tour subgraph results in different L_1^{+1} partial tour subgraphs. Configurations (1) to (6) are then added to each of these partial tour subgraphs representing the best partial tour subgraph of its corresponding equivalent class in order to receive L_1^{+2} partial tour subgraphs. Again, we determine the partial tour subgraphs representing the subgraph with a minimum sum of edge weights in its equivalent class. These subgraphs contain all vertices corresponding to requested articles in the first picking aisle.

In order to receive L_2^- partial tour subgraphs, edge configurations representing the change-over from one picking aisle to another have to be added to these L_1^{+2} partial tour subgraphs. By definition, each vertex included in an order picking tour must have an even degree. Therefore, picking aisles have to be connected by an even number of edges. Furthermore, an optimal order picking tour contains at most two edges between two vertices. This leads to 14 different edge configurations that can be used to change from one picking aisle to another (Roodbergen & de Koster, 2001b). Some of these configurations are depicted in Fig. 3 (b).

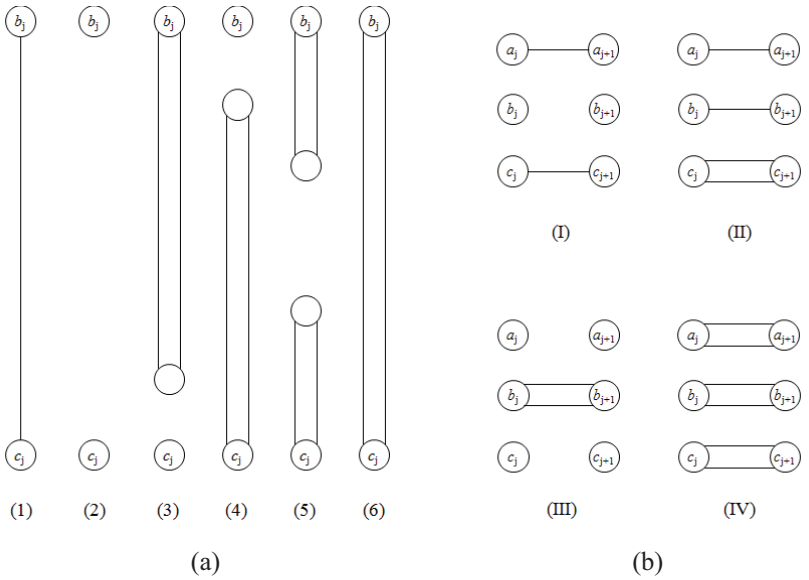


Fig. 3: (a) Possible edge configurations for traversing a subaisle
(b) Some configurations for changing between aisles

For each equivalent class, the best L_2^- partial tour subgraph is determined and configurations (1) to (6) are added to these subgraphs which results in L_2^{+1} subgraphs. Following this procedure, we finally obtain L_m^{+2} partial tour subgraphs. Now we get an optimal order picking tour by determining the partial tour subgraph which results in a minimum sum of edge weights and corresponds to a feasible order picking tour. A pseudo-code of the algorithm is given in the appendix.

An optimal order picking tour provided by this solution approach is depicted in Fig. 4. The time-complexity function of this exact solution approach is linear in the number of picking aisles m and the

number of requested articles n (Roodbergen & de Koster, 2001b) and, therefore, application of the exact algorithm to a single instance of the PRP will not result in unacceptable computing times. Even large instances can be solved within fractions of a second. However, when applied within a complex heuristic algorithm for the JOBPRP where many PRP have to be solved repeatedly, the computing time needed by the exact algorithm can become a critical issue. For example, Henn et al. (2010) have reported computing times up to 20 minutes for solving an OBP with 60 orders by applying a rank-based ant colony system, although they have used the simple s-shape and largest gap routing strategies which will be presented in the following section.

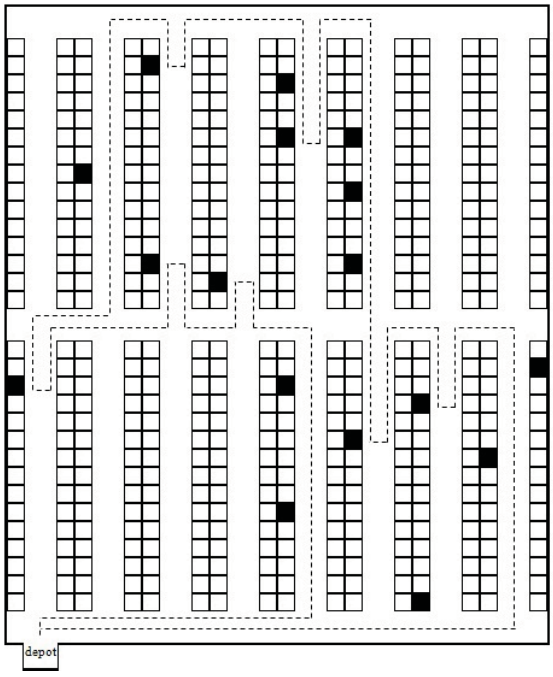


Fig. 4: Example for an optimal order picking tour

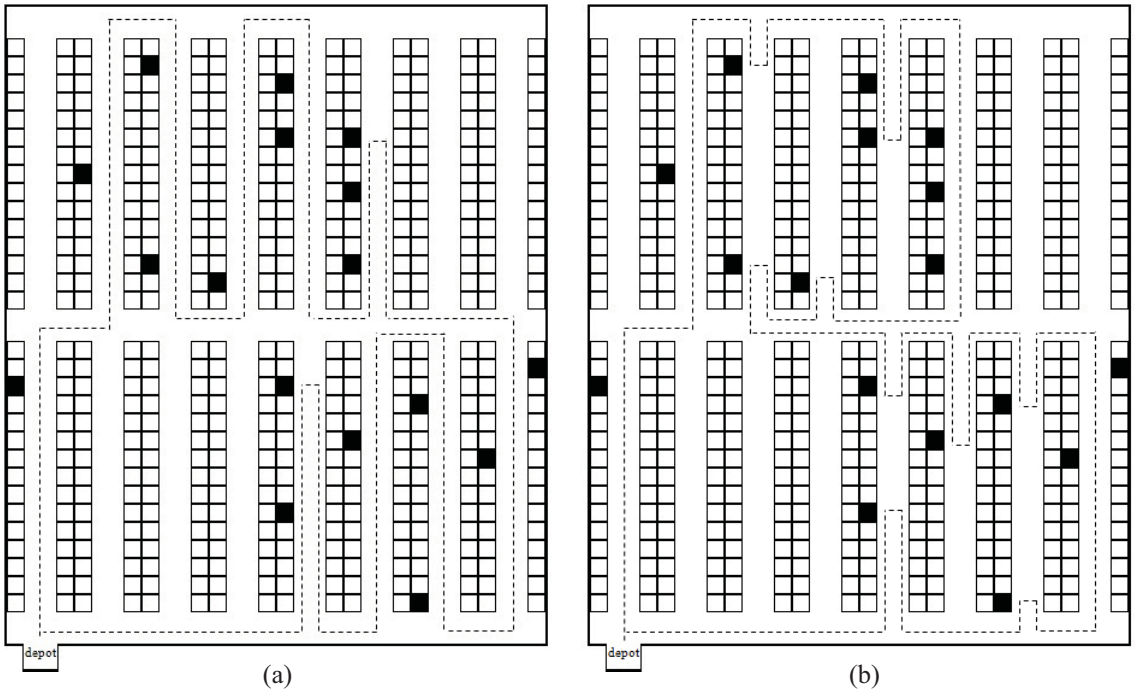
5.2 S-shape and largest gap strategy

When dealing with the OBP, the s-shape and the largest gap strategies are the most frequently used routing policies. In the following brief descriptions of these and other routing strategies, it is assumed that the depot is located in front of the leftmost picking aisle. We refer to Hall (1993) for a presentation of the strategies for warehouses with one block and to Roodbergen & de Koster (2001b) for a detailed description of the algorithms for warehouses with multiple blocks. The following denotations are used to describe the routing strategies. Let $j_{\min}$ and $j_{\max}$ be the leftmost and the rightmost picking aisle containing at least one requested article, and let $j^i_{\min}$ and $j^i_{\max}$ be the leftmost and rightmost subaisle of block i that has to be visited, respectively. The two routing strategies can be described as follows:

The s-shape (or traversal) routing policy is a simple strategy for routing order pickers. The main idea of this strategy is to traverse each subaisle containing at least one requested article. The picker starts at the depot and proceeds to $j^2_{\min}$ by traversing $j_{\min}$ of block #1. Then, from left to right, each subaisle of block #2 containing a requested article is traversed. If the number of subaisles to be traversed in block #2 is odd, the picker returns to the middle cross aisle after collecting the items located in $j^2_{\max}$. Having

completed all picks in block #2, the picker moves to $j^1_{\max}$ and starts retrieving all items to be collected from block #1 by traversing the corresponding subaisles one by one from the right to the left. As it can be the case in block #2, it may also be necessary to apply the return strategy to the last subaisle to be visited. Finally, the order picker returns to the depot.

Note that this algorithm differs slightly from the s-shape routing policy described by Roodbergen & de Koster (2001b). After having changed over from block #2 to block #1, Roodbergen & de Koster (2001b) choose the leftmost subaisle of the first block that still has to be visited instead of $j^1_{\max}$ if this subaisle is closer to $j^2_{\max}$. However, they state that this approach will increase the tour length because it may lead to routes in which the subaisles in block #1 are visited from left to right and after visiting the subaisles, a considerable part of the front cross aisle has to be traversed before reaching the depot. Therefore, we use this small modification of the s-shape policy. An example route following the s-shape policy is depicted in Fig. 5 (a). The time-complexity function is independent of the number of requested articles and increases linear in the number of picking aisles.



(a) (b)

Fig. 5: (a) Example route for the s-shape strategy
(b) Example route for the largest gap strategy

When applying the largest gap policy, each subaisle containing at least one requested article is treated in such a way that the non-traversed distance is maximal. This concept leads to the following strategy:

The order picker leaves the depot by traversing the first part of $j_{\min}$ and subaisle $j^2_{\min}$. He then moves to $j^2_{\max}$ by entering each subaisle of block #2 from the rear cross aisle up to its largest gap. Subaisle $j^2_{\max}$ is traversed and the middle cross aisle is used to collect the remaining articles located in this block. Afterwards, the picker moves to the leftmost subaisle of block #1 that has to be visited and proceeds from left to right by entering the subaisles of block #1 from the middle cross aisle up to their largest gaps. When finally $j^1_{\max}$ has been traversed, the picker returns to the depot collecting the remaining items by entering the corresponding subaisles from the front cross aisle.

The time-complexity function of this routing algorithm is linear in the number of articles and picking aisles. Again, this algorithm slightly differs from the description by Roodbergen & de Koster (2001b), where the picker – after having proceeded from block #2 to block #1 – is permitted to travel to the rightmost subaisle of block #1 that has to be entered from the middle cross aisle. This modification leads to routes with fewer direction changes since the largest gap policy by Roodbergen & de Koster (2001b) may result in routes in which the order picker continues to the right after having changed over to block #1 resulting in a direction change. Now two changes in direction are necessary in order to apply the largest gap strategy to this block, followed by a further direction change necessary to return to the depot.

An example for a route provided by our version of the largest gap strategy is given in Fig. 5 (b). It becomes evident that this route can no longer be considered as "simple" and "straightforward". This view particularly holds if the route is compared to the optimal solution to the same problem depicted in Fig. 4. Thus, we conclude again that for warehouse layouts with two or more blocks no valid argument exists for restricting the routing of the order pickers to solutions provided by simple heuristics like s-shape or largest gap strategy.

5.3 Aisle-by-aisle heuristic

As for the aisle-by-aisle heuristic by Vaughan & Petersen (1999) each picking aisle that contains at least one article to be picked is visited exactly once, i.e. the order picker starts at the depot, picks all requested articles in the first picking aisle, then he collects all articles in the second aisle and so on. After reaching the rightmost picking aisle that has to be visited, he returns to the depot. Hence, one has only to determine which cross aisle is used to change over from one picking aisle to another. This is done by means of dynamic programming.

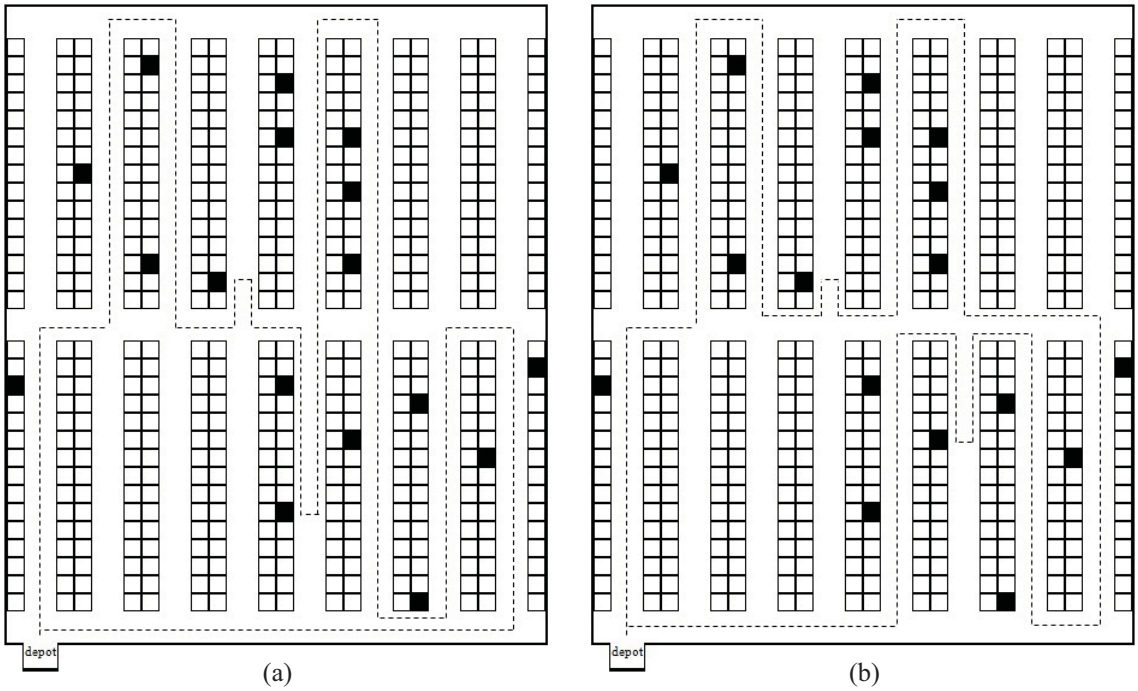


Fig. 6: (a) Example route for the aisle-by-aisle strategy
(b) Example route for the combined^{+} heuristic

At first for each cross aisle $i \in \{\text{"front"}, \text{"middle"}, \text{"rear"}\}$, the distance is determined which has to be travelled when the order picker starts at the depot, visits the storage locations of all requested articles in the first picking aisle and exits the first picking aisle by using cross aisle i. Then the second picking aisle is considered and the distance is calculated which the order picker has to travel if he exits picking aisle 1 via cross aisle $i \in \{\text{"front"}, \text{"middle"}, \text{"rear"}\}$, picks all requested articles located in the second picking aisle and then exits this aisle by using cross aisle $\tilde{i} \in \{\text{"front"}, \text{"middle"}, \text{"rear"}\}$. This results in three different distances for each cross aisle $\tilde{i}$. For each cross aisle $\tilde{i}$, the smallest of these three distances is taken as the distance needed to start at the depot, pick all requested articles located in the first two picking aisles and exit picking aisle 2 via cross aisle $\tilde{i}$. This procedure is applied to each picking aisle until $j_{\max}$ has been taken into consideration. This results in the distance to be travelled by the order picker when picking all requested articles and exiting the rightmost picking aisle through the front cross aisle. Since the picker has to end his tour at the depot, the distance between $j_{\max}$ and the depot has to be added to the covered distance. An example route following the aisle-by-aisle heuristic is depicted in Fig. 6 (a). For this heuristic, each picking aisle has to be considered once and, therefore, the time-complexity function is linear in the number of picking aisles.

5.4 Combined$^+$ heuristic

The combined heuristic introduced by Roodbergen & de Koster (2001b) follows a similar approach as the aisle-by-aisle heuristic does. Here each subaisle containing at least one requested article is visited exactly once. Picking tours are constructed by considering subaisles instead of picking aisles. Consequently, each block is considered separately. At first all requested articles located in block #2 are collected and then block #1 is considered.

The picker starts at the depot, enters the leftmost picking aisle that contains at least one requested article, and proceeds up to the middle cross aisle. Now only block #2 is considered to which the aisle-by-aisle heuristic designed for warehouses with a single-block layout is applied. Note that the picker will not return directly to the depot but complete his tour through block #2 by leaving the last subaisle to be visited in this block via the middle cross aisle. Then the picker proceeds to the rightmost subaisle that has to be visited in block #1. Again, the aisle-by-aisle heuristic for single-block layouts is applied. In contrast to the original aisle-by-aisle heuristic, the picker has to move from right to left.

In the described procedure, the routing scheme for block #2 includes a move from left to right. Thus, in block #1, the order picker has to travers the leftmost aisle with pick locations in order to access block #2. However, there may be situations in which it can be advantageous to deviate from this path (e.g. if the storage positions to be visited in the first subaisles of block #1 are located near to the middle cross aisle (Roodbergen & de Koster, 2001b)). In this case, the picker may visit several subaisles of block #1 before moving on to block #2. Generally, a route can be constructed as follows: Let m be the number of picking aisles. The order picker starts at the depot and proceeds to a picking aisle $j^* \in \{1, \ldots, m\}$ by using the front cross aisle. Then the slightly modified aisle-by-aisle heuristic explained in the previous paragraph is applied to the first j^* subaisles of block #1. After picking all requested items in these subaisles, the picker

goes to block #2 to visit all pick locations in this block. (The route through block #2 is not changed by this modification.) The modified aisle-by-aisle heuristic approach is applied to the last $m - j^*$ subaisles of block #1. Finally, the picker returns to the depot. By optimizing over $j^* \in \{1, \ldots, m\}$ we obtain a route that is not longer than the route provided by the original combined heuristic.

The combined heuristic into which this optimization step has been integrated is called combined$^+$ heuristic (Roodbergen & de Koster, 2001b). Since the combined heuristic consists of multiple applications of the aisle-by-aisle heuristic (one application for each block), the time-complexity function for this heuristic also increases linear with the number of picking aisles. When using the combined$^+$ heuristic, a slightly modified variant of the combined heuristic is applied for each $j^* \in \{1, \ldots, m\}$, generating m different routes. Therefore, we have a quadratic increase of the time complexity function in the number of picking aisles. An example route for the combined$^+$ heuristic is shown in Fig. 6 (b). For a more detailed description of this heuristic we refer to Roodbergen & de Koster (2001b).

5.5 A Heuristic derived from the exact solution approach

The time-complexity function of the exact approach is linear in the number of picking aisles m. However, it contains a large constant by which m has to be multiplied. Therefore, the application of this algorithm to problem instances of a size encountered in practice is more time-consuming than the usage of simple routing strategies. The constant originates from the number of subgraphs that have to be constructed in the exact solution approach. In each step, i.e. for each picking aisle, $6 + 6 + 14 = 26$ edge configurations have to be added to up to 25 partial tour subgraphs. This means, if we neglect the fact that the addition of some configurations to certain partial tour subgraphs cannot lead to an (optimal) order picking tour, up to $26 \cdot 25 = 650$ subgraphs have to be constructed in each iteration of the algorithm.

Therefore, in order to decrease the large constant in the time-complexity function, we do not always consider partial tour subgraphs out of all equivalent classes but, at some points, only the partial tour subgraph with the minimum sum of edge weights. To be more precise, all partial tour subgraphs except the shortest one are deleted after each change of a picking aisle. Based on this idea, a heuristic solution approach can be designed as follows:

Consider the exact algorithm at the iteration before the requested articles in picking aisle j are treated,, i.e. after having constructed the L_j^- partial tour subgraphs. Then the shortest of the L_j^- partial tour subgraphs is determined and configurations (1) to (5) (see Fig. 3) are added to this subgraph. This results in up to five L_j^{+1} partial tour subgraphs. Following the procedure of the exact solution approach, configurations are added to each partial tour subgraph representing the best subgraph of its corresponding equivalence class. Then, the same procedure is applied to the L_j^{+2} partial tour subgraphs which leads to L_{j+1}^- partial tour subgraphs. This transition includes an aisle change and, therefore, all subgraphs, except the best partial tour subgraph, are deleted.

Furthermore, as can be derived from the above description, configuration (6) is neglected, i.e. this algorithm will not lead to order picking tours in which a subaisle is traversed twice. This configuration

is of course needed to ensure optimality of the order picking tour. (Consider a PRP in which the only requested articles are located in the first subaisle of block #2. Then, an optimal solution has to contain configuration (6) in the first subaisle of block #1.) However, pretests have shown that only a few cases exist in which this configuration is necessary. When this configuration is used, it often leads to non-optimal picking tours. Also, several optimal picking tours may exist and some of these optimal tours can be obtained without using configuration (6).

Of course, optimality of an order picking tour cannot be guaranteed by this approach. Moreover, this heuristic may not lead to a feasible solution at all because the heuristic approach will only result in a few subgraphs and not all L_m^{+2} partial tour subgraphs necessarily correspond to a feasible order picking tour. Since it is possible that none of the subgraphs obtained correspond to a feasible order picking tour, this approach may lead to an infeasible solution. Therefore, the following modification is applied:

In order to guarantee that at least one feasible order picking tour is obtained, we always consider a subgraph belonging to one of the equivalence classes 3, 4 and 6 to 9 in Roodbergen & de Koster (2001b). After each step in which the shortest partial tour subgraph is determined, it is checked whether this subgraph belongs to one of these equivalent classes. If this is not the case, we additionally consider the shortest subgraph belonging to one of these classes. This is done because only eight equivalence classes (classes 2 to 9) correspond to feasible order picking tours (Roodbergen & de Koster, 2001b) and two of these may lead to an infeasible solution if they occur as $L_{j_{\max}}^{+1}$ partial tour subgraphs, where $j_{\max}$ denotes the last picking aisle to be visited. These two equivalent classes are denoted as $(0, 0, 0, 1)$ and $(0, 0, e, 1)$. The denotation $(0, 0, e, 1)$ means that each L_j partial tour subgraph belonging to this class has the characteristic that the vertices a_j and b_j have zero degree, the vertex c_j has even degree and the graph consists of one component. Hence, assuming that we have a $L_{j_{\max}}^{+1}$ partial tour subgraph belonging to the class $(0, 0, e, 1)$, i.e. only the vertex $c_{j_{\max}}$ (the vertex corresponding to the front cross aisle) is connected to the left of the subgraph, it is not possible to add edge configurations to the subaisle $j_{\max}$ of block #2 without losing the connectivity of the subgraph (if there is at least one requested article in this subaisle).

This modification guarantees feasibility of the solutions provided by the heuristic. However, adding edge configurations to only one instead of 25 subgraphs may lead to a poor solution quality. Especially, in the early stages of the algorithm decisions may be made which result in a poor final solution quality, because, at an early stage, a single edge configuration can have a quite large impact on the sum of edge weights of the subgraph. If a PRP with a small number of requested articles per subaisle is considered, then configuration (1) will lead to a larger sum of edge weights than configurations (3) to (5). Hence, the heuristic will tend to select one of these three configurations resulting in subgraphs that are not connected. Then, at a later stage, edge configurations may have to be added to connect the subgraph regardless of the impact on the length of the order picking tour. In order to overcome this problem, our heuristic is combined with the exact solution approach. This means the exact solution approach is applied to the first $s = \lceil p \cdot m \rceil$ picking aisles resulting in several L_r^{+2} partial tour subgraphs. Then the heuristic solution approach is applied to the last $m - s$ picking aisles. The parameter p denotes the percentage of picking aisles to which the exact solution approach is applied. The larger p is chosen, the higher the computational

effort gets, but a larger value of p tends to result in a better solution quality. The choice $p = 1$ results in the exact solution approach. In the appendix, a pseudo-code of this heuristic approach is given in order to point out the differences between the exact algorithm and the heuristic solution approach.

6 An iterated local search for the JOBPRP

In order to solve the JOBPRP, we combine the routing algorithms explained in section 5 with the iterated local search algorithm (ILS) of Henn et al. (2010) for the OBP. The heuristic consists of two alternating phases, namely a local search and a perturbation phase. In the local search phase an initial solution is improved by means of a certain neighborhood structure until a local optimum is reached. In order to overcome local optima, the incumbent solution is partially modified in the perturbation phase and improved in the local search phase again. If the new solution stemming from the local search procedure passes an acceptance criterion, this solution becomes the new incumbent solution. Otherwise, the perturbation phase is applied to the previous solution again.

A general pseudo-code of our algorithm for the JOBPRP is depicted below. For a more detailed presentation of the ILS algorithm we refer to Henn et al. (2010). When dealing with the joint problem, it is sufficient to consider solutions for the OBP and solve the PRP in order to evaluate the solutions. Consequently, we only deal with solutions for the OBP, i.e. each solution x does not correspond to the JOBPRP, but only to the batching (sub)problem. The solution approaches for the PRP are only used for the determination of the corresponding objective function values. Therefore, the following definition regarding the objective function is needed: Let $f_r(x)$ be the objective function value resulting from the solution x for the OBP if the routes are constructed by applying the routing algorithm r. Furthermore, let N^* denote the number of batches in the best known solution.

Algorithm 1 Pseudo-code of an ILS for the JOBPRP

Input: problem data, rearrangement parameter λ, threshold parameter μ, interval t, routing strategy r;

```
generate initial solution x by applying the FCFS rule;
x* := local_search(x); x_inc := x*;
repeat
    x := perturbation(x_inc, ⌊N* · λ + 1⌋);
    x := local_search(x);
    if f_r(x) < f_r(x*) then
        x* := x; x_inc := x;
    end if
    if no improvement of f_r(x*) during t and f_r(x) − f_r(x*) < μ · f_r(x*) then
        x_inc := x;
    end if
until termination condition is met
```

An initial (and first best) solution x is determined by applying the first-come-first-served (FCFS) rule. Then the local search procedure is used in combination with the routing strategy r in order to improve x. Afterwards, the perturbation and the local search phase alternate until a termination condition is met.

In the local search procedure two different types of neighborhoods are used. With respect to the first neighborhood structure, two solutions are called neighbors if one solution can be achieved from the other by interchanging two orders of the batches (SWAP). The second structure considers solutions that can be achieved by assigning a single order to a different batch (SHIFT). The local search phase stops when no further improvement is possible by means of these neighborhood structures. The perturbation phase constructs a new solution by interchanging a random number of orders of two batches. An iteration of this phase can be described as follows: Two batches α and β and an integer number v between 1 and half of the number of orders in α and β are randomly selected. The first v orders of α and β are removed, respectively. Then the v orders from α are assigned to β and the other way round as long as the capacity constraints are not violated. The remaining orders are assigned to a new batch. This iteration is repeated $\lfloor N^* \cdot \lambda + 1 \rfloor$ times, i.e. the more batches are contained in the best known solution, the more iterations the perturbation phase consists of.

7 Numerical Experiments

7.1 Setup

In order to evaluate the performance of the algorithms for the PRP and the combinations for solving the JOBPRP, thorough numerical experiments are carried out. The generation of the test instances is partially based on the problem sets in Henn et al. (2010) and Henn & Wäscher (2012). We do not use the same problem instances here because they considered the OBP in a warehouse with a single-block layout, whereas in this paper a two-block layout is assumed. Furthermore, some modifications are made due to the fact that the solution quality of routing strategies is dependent on the dimension of the warehouse (e.g. the number of picking aisles).

We consider a warehouse with a two-block layout and 50 storage locations in each subaisle (25 locations on each side of the subaisle). Within each subaisle, we assume two-sided picking, i.e. when a picker is located in the center of a subaisle, he is able to collect articles from both sides without additional movements. A single storage location has a length of 1 length unit (LU) and a width of 1.5 LUs. The distance between two opposite storage locations belonging to the same subaisle amounts to 2 LUs. In addition, the picker has to move 1 LU from the first or the last storage location of a subaisle to reach a cross aisle. Therefore, a picker has to travel 46 LUs to traverse a subaisle and 5 LUs to change an aisle. Since we aim to evaluate the performance of routing strategies and since the performance of the algorithms is known to be dependent on the number of picking aisles, we deviate from the problem classes by Henn & Wäscher (2012) who set the number of aisles to 10. Instead, we consider problem instances with either 10, 20 or 30 picking aisles (10 or 30 aisles for the JOBPRP) which results in a warehouse with 1000, 2000 and 3000 storage locations, respectively.

In our numerical experiments we consider problem instances with four different sizes: 20, 40, 60 and 80 customer orders. The number of requested items in a customer order is uniformly distributed over $\{5, \ldots, 25\}$. The capacity of the picking device, which is the maximum number of items that can be

collected in a single tour, amounts to 30, 45, 60 and 75. This means that the expected maximum number of customer orders included in one tour varies between 2 and 5. Of course, when dealing with the PRP, the number of orders is set to 1 and the capacity of the picking device is not smaller than the number of requested items. In the experiments for the PRP, the number of requested items in this customer order has been fixed to 30, 45, 60 and 75. Both for the PRP and for the JOBPRP we consider a random storage assignment, i.e. the probability of each article for being contained in a customer order is the same. This kind of storage assignment has also been used by Albareda-Sambola et al. (2009) and Henn et al. (2010).

A combination of the above-mentioned parameter values results in 12 problem classes for the PRP and 32 classes for the JOBPRP. For each PRP class, 100 test instances have been generated and for each class of the JOBPRP, 50 instances have been considered which leads to 2800 problem instances. The computations have been carried out on a desktop PC with a 3.4 GHz Pentium processor and 8 GB RAM. All algorithms have been encoded in C++ using Microsoft Visual Studio 2013. The problem instances can be downloaded on http://www.mansci.ovgu.de/mansci/en/ Research/Materials/2015.html. Before the JOBPRP is considered, the solution quality of the algorithms for the PRP is investigated in order to decide which algorithms are integrated into the iterated local search approach.

7.2 Picker Routing Problem

In this section, the performance of the solution approaches for the PRP presented in section 5 is evaluated. The heuristic approach in section 5.5 includes a combination with the exact algorithm and, therefore, contains a parameter p. In order to determine appropriate values for p, pretests have been carried out. Based on these tests we have chosen $p = 0.25$. The results from the numerical experiments for the PRP are depicted in Table 1. Each instance has been solved by each of the algorithms in less than one second. Computing times, therefore, are not reported in greater detail, here.

Table 1: Average deviation from optimal objective function value in %

m	n	SS	LG	AA	C^+	BCO
10	30	20.2	19.3	15.6	6.8	**4.0**
10	45	15.2	22.2	13.3	5.1	**3.2**
10	60	9.6	23.4	8.8	3.7	**2.3**
10	75	8.9	26.5	9.2	3.2	**2.9**
20	30	27.2	22.3	16.3	7.2	**5.8**
20	45	24.4	22.1	16.4	6.9	**4.5**
20	60	20.8	23.7	14.5	6.4	**3.1**
20	75	18.4	23.5	14.2	6.6	**2.7**
30	30	27.4	27.7	16.3	**7.0**	7.6
30	45	26.4	26.3	16.5	7.3	**5.4**
30	60	25.0	24.3	16.7	7.0	**4.3**
30	75	23.9	25.1	16.1	7.0	**3.8**
average		20.6	23.9	14.5	6.2	**4.1**

The objective function values obtained by the routing strategies s-shape (SS), largest gap (LG),

aisle-by-aisle (AA), combined$^+$ (C$^+$) and the heuristically modified exact approach from section 5.5 (denoted by BCO - "best class only", because only the best equivalence class of an iteration is considered) are compared to the optimal objective function value obtained by the algorithm by Roodbergen & de Koster (2001a) and the average deviation from the exact procedure is depicted. The problem classes can be identified by the number of picking aisles m and the number of requested articles n. For each problem class the number in bold represents the minimum average deviation from the optimal objective function value.

The results presented in Table 1 demonstrate that BCO outperforms the other routing strategies and leads to the shortest order picking tours. Furthermore, the solution quality improves with an increasing number of requested articles for the following reason: The exact algorithm as well as the heuristic BCO does not explicitly consider each article but rather each subaisle. Let us think of the configurations added to collect requested articles in a subaisle. The sum of edge weights that results from adding configuration (1) or (6) (see Fig. 3) is independent of the requested articles. In comparison to that, the sum of edge weights for configurations (3) to (5) is strongly dependent on the location of the requested articles within the subaisle. Therefore, the sum of edge weights may vary very strongly for a small number of requested articles but, for a large number of requested articles in a subaisle, it converges to the sum of edge weights that results from adding configuration (6). Hence, several equivalence classes may lead to a near optimal solution and the impact of deleting an equivalence class, which would lead to an optimal order picking tour, is not as large as it is for problems with a small number of requested articles. In addition, it can be seen that the number of picking aisles also has an impact on the solution quality of BCO. This is also due to the effect explained above because an increasing number of picking aisles results in a decreasing number of requested articles per picking aisle.

The s-shape heuristic, which represents the most frequently used routing policy in practice (Roodbergen, 2001), leads to poor results. For problem instances with a small number of picking aisles and a large number of articles, the s-shape heuristic provides acceptable order picking tours because if there are a lot of requested articles in each subaisle, the additional length to traverse each subaisle through the entire length compared to the distance that is needed to collect the requested articles in a subaisle will be small. However, the solution quality deteriorates with an increasing number of picking aisles, where the s-shape heuristic leads to average deviations of up to 27%. This observation coincides with the results from numerical experiments done by Roodbergen & de Koster (2001b). Over all problem classes, the average deviation amounts to 20.6%, which is more than 16 percentage points higher than the deviation of the objective function value obtained by BCO from the optimal objective function value.

Another routing strategy frequently used when dealing with the OBP, the largest gap heuristic, leads to an average deviation from the optimal total tour length of 23.9%. When solving a PRP in a warehouse with a single-block layout, it is known that the solution quality of this heuristic improves with a decreasing number of articles because in this case the distances between adjacent requested articles within the same picking aisle (called gaps) are getting larger. Since order picking tours constructed by this routing strategy do not contain the part of an aisle that corresponds to the largest gap, larger gaps lead to a better solution.

This argumentation matches with the results from the instances containing 10 picking aisles. However, if we take a look at the results corresponding to problem classes containing 30 aisles, we will see the opposite: The solution quality improves with an increasing number of requested articles. The problem instances containing 30 picking aisles are characterized by a quite small number of requested articles per subaisle. This number ranges from 0.5 to 1.25 and, therefore, it can be expected that the gaps between requested articles in a subaisle are quite large. Thus, the largest gap policy leads to very short distances covered in subaisles. The distance covered in cross aisles, however, may be quite large. In a worst case scenario, the middle cross aisle has to be traversed twice which results in large distances for problems with a large number of picking aisles. If the number of requested articles gets larger, the length of an order picking tour will increase, especially the distances covered in the subaisles. Therefore, the percentage of the distance covered in cross aisles decreases and the large distances resulting from the movements in the middle cross aisle have a smaller impact on the objective function value which leads to a smaller deviation from the optimal objective function value. However, if the number of requested articles is rising, the deviation from the optimal objective function value will increase at some point due to the increasing deviation resulting from the movements within the subaisles.

Apart from the two heuristics most commonly used for solving the PRP in single-block warehouses, we have tested the performance of the aisle-by-aisle and the combined$^+$ heuristic, which have been specifically developed for solving PRPs arising in warehouses with multiple blocks. Across all problem classes, the aisle-by-aisle heuristic leads to an average deviation of 14.5% and, therefore, also results in quite long order picking tours and is outperformed by the combined$^+$ heuristic. This is a consequence of the fact that the combined$^+$ heuristic considers the problem blockwise, i.e. only subaisles are considered instead of complete picking aisles. Therefore, the combined$^+$ heuristic leads to an acceptable solution quality even for problem instances in which the number of requested articles per subaisle is quite small. Nevertheless, the solution quality deteriorates with a decreasing number of requested articles per subaisle because this heuristic is a combination of the s-shape and the return strategy and, as mentioned before, the s-shape policy leads to very poor results in these cases.

In summary, it can be stated that BCO leads to the smallest deviations from the optimal objective function value. Furthermore, we can see that the combined$^+$ heuristic also results in rather short order picking tours. However, if only single instances of the PRP have to be solved, the exact approach should be applied because the computing times are below one second and, therefore, negligible. Computing times become an issue, though, when instances of the PRP have to be solved repeatedly within the proposed ILS approach to the JOBPRP, as will be demonstrated in the next subsection.

7.3 Joint Order Batching and Picker Routing Problem

In order to solve the JOBPRP, several routing strategies are integrated into the iterated local search approach presented in section 6. As routing strategies we used the exact solution approach, the s-shape and largest gap policy, the combined$^+$ heuristic and BCO which have been shown to perform quite well in the previous subsection. The parameters for the iterated local search have been chosen according to

Henn et al. (2010) and are, therefore, set as follows: $\lambda = 0.3$, $\mu = 0.05$, $t = \frac{T}{10}$, where T denotes the time limit that is used as the termination criterion. The time limit $T = 3 \cdot N$ is measured in seconds and is only dependent on the number of orders, i.e. on the problem size of the OBP and not on the routing strategy used. Thus, there is a tradeoff between considering a larger part of the solution space of the OBP by running a large number of iterations in the iterated local search and constructing good routes. This can also be seen in Table 2.

Table 2: Average number of iterations performed by the ILS algorithm ($C = 30$)

m	N	SS	LG	C^+	BCO	Exact
10	20	9220	3822	762	292	181
10	40	3145	1058	201	96	60
10	60	1251	455	116	34	21
10	80	717	206	36	12	7
30	20	3481	1861	296	122	81
30	40	1117	543	66	29	20
30	60	376	174	20	11	6
30	80	209	97	9	6	3

In this table, the number of iterations performed by the ILS is presented in dependency of the number of picking aisles m, the number of orders N and the routing strategy. An iteration includes two consecutive local search and perturbation phases. Of course, the number of iterations decreases with an increasing number of picking aisles and orders. However, the number of iterations also decreases significantly if more complex routing strategies are used. This is because a large number of PRPs have to be solved in each local search phase in order to evaluate the solutions. Although only fractions of a second are needed to solve a single PRP to optimality, the routing algorithm used in the ILS has a large impact on the number of different solutions for the OBP that can be considered. Therefore, it has to be investigated if this low number of iterations can be compensated by finding good solutions for the PRPs.

In Table 3 the average deviation of the objective function value obtained by the application of the ILS in combination with a routing heuristic from the objective function value of the ILS with the exact routing algorithm (denoted by ILS-Exact) is depicted for each problem class. Due to the fact that all deviations are positive, we can conclude that the combination of the ILS and the exact routing algorithm outperforms the application of heuristic routing approaches in the ILS for the OBP. When dealing with the OBP, it is very common to use the s-shape or the largest gap strategy in order to evaluate solutions (Albareda-Sambola et al., 2009; Henn et al., 2010; Henn & Wäscher, 2012). The usage of these simple routing strategies, however, results in objective function values far above the objective function values that can be obtained by combining a metaheuristic for the OBP with the exact routing algorithm. On average, across all problem classes, the common integration of the s-shape and the largest gap strategy into the batching algorithm leads to tours that are 17.95% and 22.48% longer than tours obtained by ILS-Exact. If the number of picking aisles is small ($m = 10$) and the capacity of the picking device is large ($C = 75$), then the integration of the s-shape policy leads to deviations that are below 10%. This is because a larger capacity results in batches containing more articles and due to the fact that the s-shape

strategy leads to good solutions if the number of articles to be picked per aisle is quite large. However, in all other problem classes, and especially if the number of picking aisles is large ($m = 30$), both the s-shape and the largest gap strategy result in very poor solutions.

The integration of the more complex routing algorithms combined$^+$ and, in particular, BCO lead to acceptable results with an average deviation of 4.87% and 2.64%, respectively. Thus, it can be seen that the routing strategy used in the batching algorithm has a large impact on the solution quality and although the number of iterations in the ILS becomes much smaller if more complex routing algorithms are used, the solution quality increases. However, it would be quite simple to improve the quality of the solutions obtained by ILS and an arbitrary routing heuristic. An optimization of the tours at the end of the ILS is all it takes. The ILS terminates with a best found solution in which the N customer orders are grouped into N^* different batches, resulting in N^* tours that have been constructed by applying a certain routing algorithm. The objective function value now could be improved by taking these N^* batches and solving the resulting PRPs to optimality. Since a single PRP can be solved within fractions of a second and the number of batches N^* is not larger than the number of orders N, this tour optimization can be done without consuming much additional computing time. The resulting deviations of the tour lengths from the objective function value obtained by ILS-Exact are depicted in Table 4.

After the application of the exact routing algorithm at the end of the ILS approach, the deviations from the objective function value obtained by ILS-Exact significantly decrease. If the simple s-shape and largest gap strategy are used in particular, the optimization of the tours at the end of the batching algorithm is pivotal and results in a decrease of the deviation by 16 and 20 percentage points, respectively. Nevertheless, the deviations remain positive and, therefore, indicate that the integration of the exact algorithm into the ILS is superior to using routing heuristics and optimizing the tours only at the end of the ILS. On average, the common approach for solving the OBP leads to tours 1.86% and 1.53% longer than the tours generated by ILS-Exact. The largest deviation can be observed for problem classes with a small number of picking aisles ($m = 10$) and a large capacity of the picking device ($C \in \{60, 75\}$) if the s-shape strategy is used. In these problem classes the deviation amounts to approximately 3%. However, the deviations are getting smaller if the number of picking aisles increases. In problem classes characterized by a large number of picking aisles ($m = 30$) and a small capacity ($C = 30$), the integration of the s-shape policy leads to objective function values very close to the tour lengths obtained by ILS-Exact. This is caused by the chosen time limit which is only dependent on the problem size of the OBP and not on the number of picking aisles or on the routing algorithm. Therefore, only a very small part of the solution space can be investigated if the number of picking aisles is large and the exact routing algorithm is applied. Thus, it can be expected that the deviations in Table 4 will get larger if the time limit is increased as done by Henn et al. (2010) who have permitted computing times of up to 20 minutes for OBPs containing 60 customer orders. In this case, the integration of the exact routing algorithm into the corresponding batching algorithm would become even more important.

Table 3: Average deviation from the objective function value obtained by the ILS in combination with the exact routing algorithm [%]

m	N	C	ILS			
			SS	LG	C^+	BCO
10	20	30	20.28	17.90	6.37	**3.72**
10	20	45	14.93	20.21	4.85	**1.64**
10	20	60	12.21	22.72	4.21	**1.18**
10	20	75	9.93	24.68	3.63	**0.88**
10	40	30	19.29	17.56	5.92	**3.44**
10	40	45	13.77	19.32	4.26	**1.68**
10	40	60	10.61	21.88	3.79	**1.35**
10	40	75	8.22	24.09	2.89	**1.20**
10	60	30	18.44	17.68	5.23	**2.76**
10	60	45	13.45	19.51	3.83	**1.72**
10	60	60	10.10	21.30	2.87	**1.21**
10	60	75	7.88	23.61	2.45	**0.85**
10	80	30	18.31	17.49	5.39	**2.86**
10	80	45	13.30	19.38	4.11	**1.74**
10	80	60	10.23	21.40	3.51	**1.57**
10	80	75	7.85	23.56	2.82	**1.52**
30	20	30	24.10	25.52	5.94	**4.95**
30	20	45	24.55	24.49	5.94	**3.18**
30	20	60	23.45	24.56	5.87	**2.57**
30	20	75	22.22	24.39	5.58	**2.11**
30	40	30	24.21	25.03	5.87	**4.73**
30	40	45	24.18	23.98	5.73	**3.44**
30	40	60	22.87	24.08	5.80	**3.15**
30	40	75	21.22	24.02	5.46	**2.48**
30	60	30	23.46	24.89	5.67	**4.76**
30	60	45	23.15	24.01	5.56	**3.86**
30	60	60	22.49	23.57	5.00	**3.50**
30	60	75	21.52	23.01	5.13	**2.42**
30	80	30	23.37	24.62	5.77	**4.30**
30	80	45	22.60	23.35	5.30	**3.22**
30	80	60	21.87	23.73	5.64	**3.31**
30	80	75	20.39	23.68	5.38	**3.14**
average			17.95	22.48	4.87	**2.64**

Table 4: Average deviation after application of the exact routing algorithm [%]

m	N	C	ILS			
			SS	LG	C^+	BCO
10	20	30	1.03	0.92	**0.72**	0.96
10	20	45	2.45	1.64	0.95	**0.66**
10	20	60	2.75	2.06	1.20	**0.63**
10	20	75	3.06	1.87	1.12	**0.49**
10	40	30	1.17	0.89	**0.73**	1.44
10	40	45	2.33	1.69	**0.94**	1.00
10	40	60	2.94	2.09	1.37	**0.94**
10	40	75	2.79	2.20	1.17	**0.92**
10	60	30	1.26	0.96	**0.56**	1.02
10	60	45	2.75	1.60	**0.93**	1.12
10	60	60	2.85	1.43	**0.79**	0.89
10	60	75	3.09	1.86	0.93	**0.59**
10	80	30	0.97	0.80	**0.43**	1.18
10	80	45	2.70	1.65	**1.09**	1.18
10	80	60	3.26	1.94	1.37	**1.33**
10	80	75	3.04	1.94	**1.26**	1.32
30	20	30	0.23	0.59	**0.07**	1.11
30	20	45	1.20	1.74	**0.83**	1.42
30	20	60	1.85	1.78	**0.99**	1.24
30	20	75	1.75	1.85	**0.75**	0.88
30	40	30	**0.15**	0.28	0.24	1.13
30	40	45	1.48	1.64	**0.77**	1.75
30	40	60	1.87	2.13	**1.14**	1.85
30	40	75	2.04	2.31	**1.21**	1.48
30	60	30	**0.06**	0.27	0.26	1.72
30	60	45	1.42	1.87	**0.84**	2.37
30	60	60	1.46	1.71	**0.49**	2.20
30	60	75	1.74	1.45	**0.80**	1.24
30	80	30	**0.04**	0.27	0.38	1.36
30	80	45	1.05	1.40	**0.68**	1.88
30	80	60	2.17	2.12	**1.49**	2.30
30	80	75	2.50	2.18	**1.48**	2.34
average			1.86	1.53	**0.87**	1.31

8 Conclusion and Outlook

In this paper, we dealt with the Joint Order Batching and Picker Routing Problem which is very important for an efficient operation of manual order picking systems. For the solution of this problem, different solution approaches for the Picker Routing Problem have been integrated into an iterated local search algorithm for the Order Batching Problem. Up to now, simple routing policies have been combined with sophisticated algorithms for the Order Batching Problem in order to solve this problem. However, when dealing with two-block warehouses, even these simple routing strategies may result in quite confusing routes with several changes in direction. The intention of this paper was to investigate whether the solution quality can be improved by applying more complex algorithms to the Picker Routing Problems to be solved in a batching algorithm to evaluate solutions. Therefore, different existing routing algorithms have been presented. Furthermore, a new solution approach derived from an exact routing algorithm has been designed in order to close the gap between the complex exact routing algorithm and the simple routing heuristics. Based on extensive numerical experiments with fixed computing times, the performance of the routing algorithms as well as the performance of the iterated local search approach combined with different routing algorithms have been evaluated.

The results from the numerical experiments have shown that common approaches are outperformed by far. On average, the resulting tours turned out to be up to 25% longer than the tours obtained by the iterated local search into which the exact routing algorithm has been integrated. Furthermore, it has been shown that even the solution quality of common approaches can be significantly improved by optimizing the tours after the termination of the batching algorithm. So it is pivotal to apply the exact routing algorithm at least at the end of the batching algorithm. However, if this algorithm has to be used after the batching algorithm, then it is reasonable to integrate this algorithm into the iterated local search in order to achieve further improvements without increasing the computing times.

Further research could concentrate on the consideration of picker blocking aspects because congestion is an important topic in warehouses with narrow aisles and the savings reached by the integration of the exact routing algorithm may diminish if this leads to picker blocking on a larger scale. From a practical point of view it would also be reasonable to consider the on-line variant of the joint batching and routing problem in which not all customer orders are known in advance.

References

Albareda-Sambola, M.; Alonso-Ayuso, A. & de Blas, C. (2009): Variable Neighborhood Search for Order Batching in a Warehouse. *Asia-Pacific Journal of Operational Research* 26, 655-683.

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

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

Coyle, J. J.; Bardi, E. J. & Langley, C. J. (1996): The Management of Business Logistics. 6th ed., West Publishing Company: St. Paul.

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.

de Koster, R.; Roodbergen, K. J. & van Voorden, R. (1999a): Reduction of Walking Time in the Center of de Bijenkorf. *New Trends in Distribution Logistics*, Speranza, M. G. & Stähly P., (eds.), 215-234, Springer: Berlin.

de Koster, R.; van der Poort, E. & Wolters, M. (1999b): Efficient Orderbatching Methods in Warehouses. *International Journal of Production Research* 37, 1479-1504.

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

Elsayed, E. A. & Unal, O. I. (1989): Order Batching Algorithms and Travel-Time Estimation for Automated Storage/Retrieval Systems. *International Journal of Production Research* 27, 1097-1114.

Frazelle, E. (2002): World-Class Warehouse and Material Handling. McGrawHill: New York.

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.

Goetschalckx, M.; & Ratliff, H. D. (1988): Order Picking in an Aisle. *IIE Transactions* 20, 53-62.

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.

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

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.; Dörner, K.; Strauss, C. & Wäscher, G. (2010): Metaheuristics for the Order Batching Problem in Manual Order Picking Systems. *BuR - Business Research* 3, 82-105.

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. & Wäscher, G. (2012): Tabu Search Heuristics for the Order Batching Problem in Manual Order Picking Systems. *European Journal of Operational Research* 222, 484-494.

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.

Petersen, C. G. (1997): An Evaluation of Order Picking Routeing Policies. *International Journal of Operations and Production Management*, 17, 1098-1111.

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.

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.

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.

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.

Vaughan, T. S. & Petersen, C. G. (1999): The Effect of Warehouse Cross Aisles on Order Picking Efficiency. *International Journal of Production Research* 37, 881-897.

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

Appendix

Pseudo-codes for the exact algorithm and the heuristic derived from this approach

Algorithm 2 Pseudo-code for the algorithm by Roodbergen & de Koster (2001a)

Input: problem data (containing number of picking aisles m);

compute sum of edge weights for each configuration and picking aisle;
construct L_1^{+1}-PTS by adding configurations for subaisle 1 of block #1 to an empty graph;
for equivalence classes $i = 1$ to 25 **do**
  determine the L_1^{+1}-PTS of class i with the smallest sum of edge weights;
end for
for equivalence classes $i = 1$ to 25 **do**
  construct L_1^{+2}-PTS by adding configurations for subaisle 1 of block #2 to L_1^{+1}-PTS of class i;
end for
for equivalence classes $i = 1$ to 25 **do**
  determine the L_1^{+2}-PTS of class i with the smallest sum of edge weights;
end for
for picking aisles $j = 2$ to m **do**
  for equivalence classes $i = 1$ to 25 **do**
    construct L_j^{-}-PTS by adding configurations for movements between picking aisles $j-1$ and j to L_{j-1}^{+2}-PTS of class i;
  end for
  for equivalence classes $i = 1$ to 25 **do**
    determine the L_j^{-}-PTS of class i with the smallest sum of edge weights;
  end for
  for equivalence classes $i = 1$ to 25 **do**
    construct L_j^{+1}-PTS by adding configurations for subaisle j of block #1 to L_j^{-}-PTS of class i;
  end for
  for equivalence classes $i = 1$ to 25 **do**
    determine the L_j^{+1}-PTS of class i with the smallest sum of edge weights;
  end for
  for equivalence classes $i = 1$ to 25 **do**
    construct L_j^{+2}-PTS by adding configurations for subaisle j of block #2 to L_j^{+1}-PTS of class i;
  end for
  for equivalence classes $i = 1$ to 25 **do**
    determine the L_j^{+2}-PTS of class i with the smallest sum of edge weights;
  end for
end for
out of classes $2, 3, \ldots, 9$, determine the L_m^{+2}-PTS with the smallest sum of edge weights;

Algorithm 3 Pseudo-code for the heuristic derived from the exact solution approach

Input: problem data (containing number of picking aisles $\tilde{m}$), percentage p;

compute sum of edge weights for each configuration and picking aisle;
apply the algorithm by Roodbergen & de Koster (2001a) with $m := \lceil p \cdot \tilde{m} \rceil$;
for pickings aisles $j = \lceil p \cdot \tilde{m} \rceil + 1$ to $\tilde{m}$ **do**
  for equivalencees class $i = 1$ to 25 **do**
    construct L_j^--PTS by adding configurations for movements between picking aisles $j-1$ and j to L_{j-1}^{+2}-PTS of class i;
  end for
  determine the L_j^--PTS (denoted by L_j^*) with the smallest sum of edge weights;
  construct L_j^{+1}-PTS by adding configurations for subaisle j of block #1 to L_j^*;
  if L_j^* does not belong to any of the classes $3, 4, 6, 7, 8$ or 9 **then**
    out of classes $3, 4, 6, 7, 8$ and 9, determine the L_j^--PTS (denoted by L_j^{**}) with the smallest sum of edge weights;
    construct additional L_j^{+1}-PTS by adding configurations for subaisle j of block #1 to L_j^{**};
  end if
  for equivalence classes $i = 1$ to 25 **do**
    determine the L_j^{+1}-PTS of class i with the smallest sum of edge weights;
  end for
  for equivalence classes $i = 1$ to 25 **do**
    construct L_j^{+2}-PTS by adding configurations for subaisle j of block #2 to L_j^{+1}-PTS of class i;
  end for
  for equivalence classes $i = 1$ to 25 **do**
    determine the L_j^{+2}-PTS of class i with the smallest sum of edge weights;
  end for
end for
out of classes $2, 3, \ldots, 9$, determine the L_m^{+2}-PTS with the smallest sum of edge weights;

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