WORKING PAPER SERIES

Order Batching in Order Picking Warehouses: A Survey of Solution Approaches

Sebastian Henn/Sören Koch/Gerhard Wäscher

Working Paper No. 01/2011

Impressum (§ 5 TMG)

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

Verantwortlich für diese Ausgabe:
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

Sebastian Henn • Sören Koch • Gerhard Wäscher

Order Batching in Order Picking Warehouses: A Survey of Solution Approaches

January 2011

Abstract: 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. Of all warehouse operations, order picking is considered to include the most cost-intensive ones. Even though there have been different attempts to automate the picking process, manual order picking systems are still prevalent in practice. This article will focus on order batching, one of the main planning issues in order picking systems. Order Batching has been proven to be pivotal for the efficiency of order picking operations. With respect to the availability of information about the customer orders, order batching can be distinguished into static batching and dynamic batching. Improved order batching reduces the total picking time required to collect the requested articles. According to experience from practice, this can result in significant savings of labor cost and into a reduction of the customer order's delivery lead time.

The aim of this contribution is to provide comprehensive insights into order batching by giving a detailed state-of-the-art overview of the different solution approaches which have been suggested in the literature. Corresponding to the available publications, the emphasis will be on static order batching.

In addition to this, the paper will also review the existing literature for variants and extensions of static order batching (e.g. due dates, alternative objective functions). Furthermore, solution approaches for dynamic order batching problems (like time window batching) will be presented.

Keywords: Order picking system, Order batching, Travel time minimization

Corresponding author:

Dipl.-Math. oec. Sebastian Henn
Otto-von-Guericke-University Magdeburg
Faculty of Economics and Management
- Management Science -
Postfach 4120
39016 Magdeburg
{sebastian.henn@ovgu.de}

Contents

1. Introduction

Order picking is a warehouse function dealing with the retrieval of articles from their storage locations in order to satisfy a given demand specified by customer requests (Petersen & Schmenner, 1999). Order picking arises because incoming articles are received and stored in (large-volume) unit loads while (internal or external) customers order small volumes (less-than-unit loads) of different articles. It is a function critical to each supply chain, since underperformance results in an unsatisfactory customer service (long processing and delivery times, incorrect shipments) and high costs (labour cost, cost of additional and/or emergency shipments).

Of all warehouse operations, order picking is considered to include the most cost-intensive ones. According to Frazelle (2002) up to 50% of the total warehouse operating costs can be attributed to order picking. Drury (1988, also see Tompkins et al., 2003) and Coyle et al. (1996) even estimate these costs up to 60% and 65%, respectively. The large proportion of order picking (operations) costs originates from the fact that order picking systems still involve the employment of human operators on a large scale, despite that there have been various attempts to automate the picking process.

Among such manual order picking systems, picker-to-parts systems can be considered as the most important ones, where order pickers move through the warehouse and collect the requested articles (Wäscher, 2004). For an efficient organization of the corresponding picking operations, order batching, i.e. the grouping of customer orders into picking orders, has been proven to be pivotal (de Koster et al., 1999a).

The aim of this contribution is to provide a comprehensive state-of-the-art review of solution approaches for order batching in picker-to-part systems. The focus of the paper will be on static order batching, which is based on the assumption that all customer orders are known in advance. Furthermore, this paper will also review the existing literature for variants and extensions of static order batching (e.g. wave picking, batching and sequencing problems). Finally, solution approaches for dynamic order batching problems (e.g. time window batching) will be presented.

The remainder of this paper is organized as follows: In the next section we give a brief introduction into order picking systems, the corresponding planning issues and objectives. Section 3 describes the (static) order batching problem. Furthermore, a mathematical model for this problem and an exact solution approach are presented. Since the exact solution approach is limited to problems of small size, the application of heuristic solution approaches is necessary. The sections 4 and 5 are dedicated to these heuristic approaches, Section 4 to constructive solution approaches and Section 5 to the application of metaheuristics to the order batching problem. The performance of these algorithms is discussed in Section 6. In addition to the static order batching several variants of the problem will be presented. In Section 7 wave picking will be discussed, whereas Section 8 reviews solution approaches for batching and sequencing problems. Dynamic order batching is described in Section 9. The paper concludes with a summary.

2. Fundamentals

2.1 A Classification of Order Picking Systems

Two kinds of order picking systems can be distinguished in practice (cf. Fig. 1.), namely manual order picking systems which employ human operators, and technical systems, in which the process of retrieving articles from the warehouse is completely automated. The first group of systems can be further differentiated into picker-to-parts systems and parts-to-picker systems.

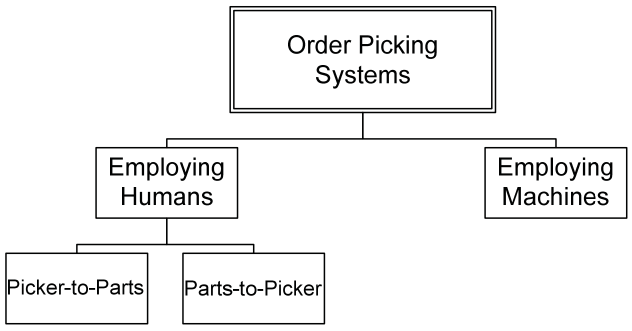

Fig. 1. Classification of order picking systems (based on de Koster et al., 2007)

In picker-to-parts systems the order picker walks or rides (e.g. on an Automated Guided Vehicle) through the picking area, stops at the storage locations of the respective articles, and removes the required number of article units / items. In low-level picker-to-parts systems the items can be removed from pallets or bins placed on the warehouse floor, or from low-level racks which are directly accessible by the order picker from the warehouse floor. In high-level (or man-aboard) systems the picking area consists of high storage racks, and a crane or a vehicle with a hoisting platform moves the order picker to the storage locations from which items have to be picked.

In parts-to-picker systems automated storage and retrieval systems (AS/RS) retrieve unit loads (pallets or bins) from the warehouse and deliver them at a transfer site (depot) where one or several stationary order pickers are located. The order pickers remove the requested items, and the AS/RS returns the unit load to its location in the warehouse.

According to de Koster et al. (2007), more than 80% of all order picking systems in Western Europe are low-level picker-to-parts systems. Therefore, in this paper we will concentrate on this type of order picking system.

2.2 Standard Layouts of Manual Order Picking Systems

In a standard layout of a (low-level) picker-to-parts system, the storage locations (bays) are of identical size. The bays are arranged on both sides of straight picking aisles of equal length and width, which run in parallel to each other and perpendicular to the front of the picking area. The depot or input/output (I/O) point is the place where the order pickers enter the picking area and where they return to in order to deposit the picked items. Order pickers are enabled to change from one picking aisle to another by means of two cross aisles, one at the front and one at the rear of the picking area. The conditions described here constitute a so-called single block layout which is depicted in Fig. 2. Depending on the types of articles, their sizes, weights, demands etc. the introduction of additional cross aisles might increase the efficiency of the system, resulting in a multi-block layout. Such layouts and other, non-standard layout types of layouts will not be discussed here any further; instead, we refer to the literature (Pohl et al., 2009).

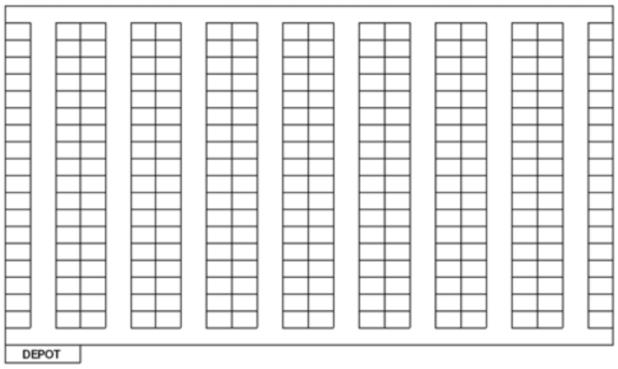

Fig. 2. Single block layout

2.3 Picking Devices and Pick Lists

When covering the distances within the picking area on foot, order pickers typically utilize devices like roll pallets or carts, which they pull or push along with them through the warehouse and on which they deposit the picked items until they finally return to the depot. Likewise, when travelling on a vehicle, there will be space available on the vehicle for intermediate storage where the picked items can be placed. Consequently, the required items are collected on tours through the warehouse, where 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.

On their tours through the warehouse, order pickers are guided by pick lists. A pick list comprises a set of order lines, each one identifying a particular article, the quantity of this article requested by a customer and the respective storage location. The order lines are already sorted into the sequence according to which the order picker is meant to collect the items.

2.4 Operative Planning Issues in Manual Order Picking Systems

Given a picking area with a fixed layout, the following essential planning issues can be identified (de Koster et al., 2007):

- *storage assignment*, i.e. the assignment of articles to storage locations;
- *zoning*, i.e. the establishment of (work) zones, to which order pickers are restricted in their picking operations;
- *order consolidation*, i.e. the transformation of customer orders into picking orders;

- *picker routing*, i.e. the determination of sequences according to which the items have to be picked and the identification of the corresponding paths in the warehouse.

Storage assignment and zoning represent medium-term planning issues and rather are part of the tactical planning level, while order consolidation and picker routing refer to the operative level in the first place.

Order consolidation can be organized in two different ways. As for discrete order picking (pick-by-order), each tour comprises the items of a single customer order only while for batch picking (pick-by-batch) items of several customer orders can be collected simultaneously on a single tour. In the latter case, the process of grouping a set of customer orders into picking orders is referred to as order batching.

With respect to the availability of information concerning customer orders, order batching can be distinguished into static (off-line) batching and dynamic (on-line) batching (Yu & de Koster, 2009). In the static case it is assumed that the set of customer orders is self-contained and complete information about its composition (i.e. for each customer order the corresponding order lines are known) is available when the batching decision is taken. In the dynamic case customer orders arrive at different points in time while the picking process is already being executed.

Picker routing deals with the determination of the sequence in which the items of a given picking order are to be picked and the identification of the corresponding (shortest) tour for the order picker which connects the respective article locations among each other and with the depot. Ratliff & Rosenthal (1983) have presented an exact (polynomial-time) algorithm for this problem, which is hardly ever used in practice, though. Order pickers do not seem to be willing to follow the routes provided by the algorithm, because of their not always straightforward and sometimes even confusing routing schemes (de Koster et al., 1999a). Instead, so-called routing strategies are applied, which may be looked upon as heuristic solution methods, not necessarily giving tours of minimal length but of plausible patterns, easy to memorize and easy to follow. In this way, the risk of missing an item to be picked is reduced, which may be an aspect more important than a small reduction of the tour length.

In Fig. 3, several customary routing strategies (Return, S-Shape, Largest Gap and Combined routing) are depicted. The black rectangles symbolize the storage locations from which items have to be picked (pick locations). When proceeding according to the Return strategy, the order picker enters each aisle in which an item has to be picked from the front cross aisle, walks up to the most distant pick location in this aisle and then returns to the front cross aisle. As for S-Shape routing, the order picker successively traverses each aisle entirely if it contains at least one pick location. Correspondingly, the first aisle is entered from the front cross aisle, the second one from the rear cross aisles, etc. With Largest Gap routing the aisles are entered from front and back aisle in such a way that the non-traversed distance between two adjacent pick locations is maximal. Only the leftmost and the rightmost aisle which contain items to be picked are traversed entirely. The Combined strategy integrates elements of the S-Shape and Return strategy. Aisles may be traversed entirely or may be entered and left from the same cross aisle. The respective solutions are usually provided by application of dynamic programming.

We note that – from a planning-theoretical point of view – it would be desirable to solve the order consolidation problem and the picker routing problem simultaneously. Such an approach, however, does not appear to be very realistic for practical purposes, due to the complexity and the size of the problem which would have to be solved (Wäscher, 2004). In practice, decisions are made sequentially (de Koster et al., 2007). Here, we will assume that a decision has already been made in favour of a particular routing strategy, which will serve as input for a subsequent order batching decision.

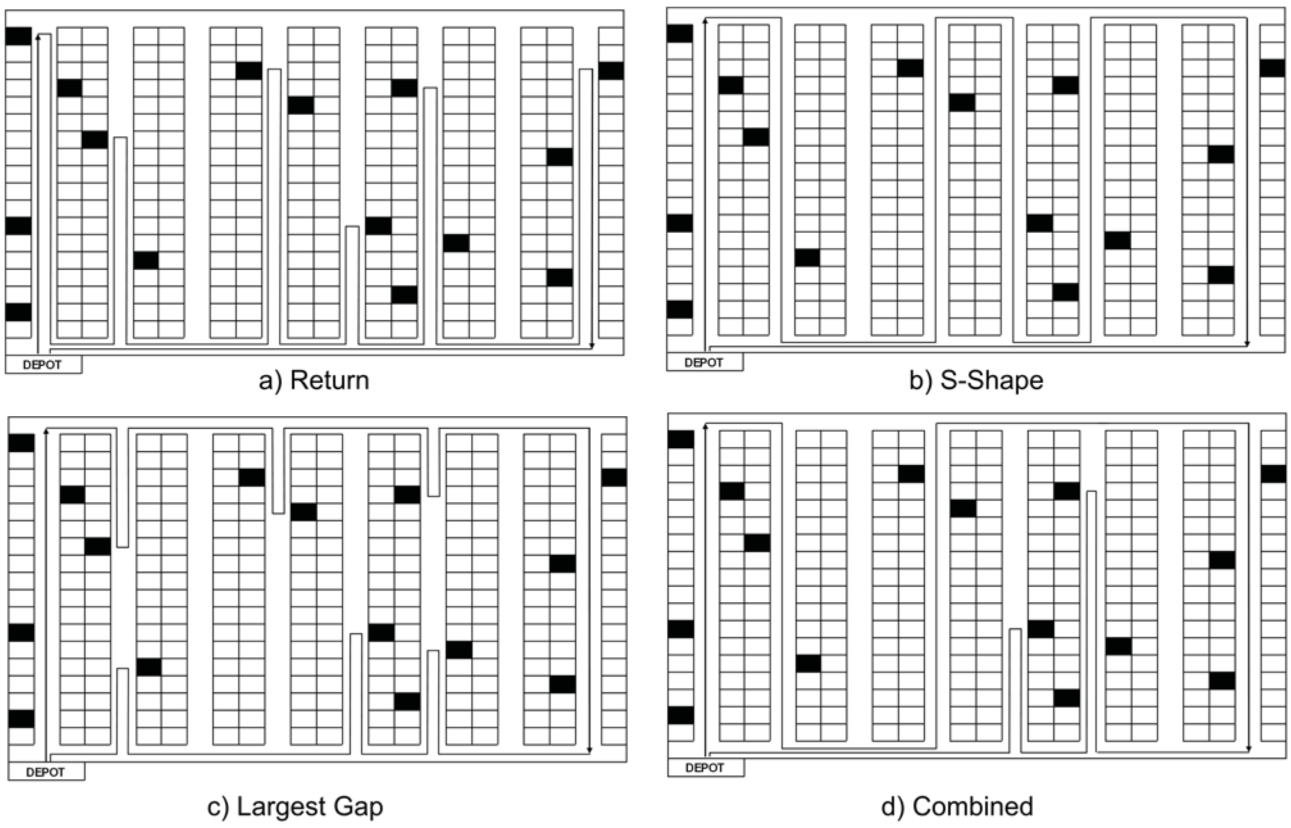

Fig. 3. Routing strategies in a single block layout (Hall, 1993; Petersen & Schmenner, 1999; Roodbergen & de Koster, 2001)

2.5 Planning Objectives of Operative Planning

Involving a large proportion of time-consuming manual operations, order picking is considered to be the most labour-cost-intensive function in a warehouse (Drury, 1988). According to experience from practice, a reduction of picking times results in significant savings of labour cost, since it does not only allow for reducing the necessary regular working hours of the pickers, but also for reducing expensive overtime or even for downsizing the (picker) workforce. Furthermore, since the picking time is an integral part of the delivery lead time, a reduction of the picking times may also immediately result in an improvement of the customer service provided by the warehouse (Henn et al., 2010). Consequently, the minimization of picking times is of vital importance for controlling the picking processes in a warehouse efficiently.

Given a particular picking order, the time the order picker takes for the completion of a tour on which the respective items are collected, will be called (*picking*) *order processing time*. Essentially, it consists of the following components:

- *travel time*, i.e. the time the order picker spends travelling from the depot to the first pick location, between the pick locations and from the last pick location to the depot;
- *search time*, i.e. the time required for the identification of articles;
- *pick time*, i.e. the time needed for moving the items from the corresponding article location onto the picking device;
- *setup time*, i.e. the time consumed by administrative and set-up tasks at the beginning and end of each tour (Chew & Tang, 1999), including the receipt of the pick list and of an empty picking device at the beginning of a tour and the return of the picking device at the depot (van Nieuwenhuyse & de Koster, 2009).

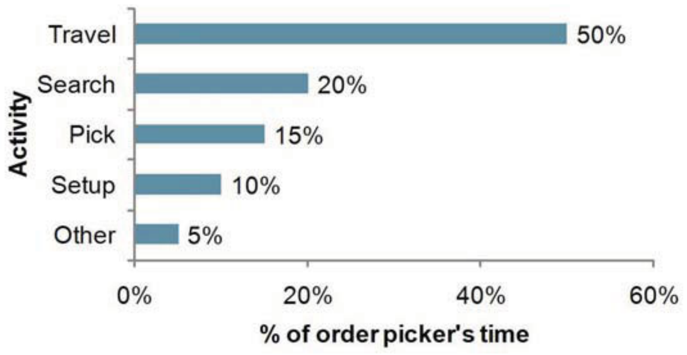

Fig. 4. Typical composition of the order processing time (Tompkins et al., 2003)

Fig. 4 depicts how the order processing time is typically composed of these time elements in practice. Among these components, the travel time consumes the major proportion. The other components can either be looked upon as constants (search times and pick times) or as neglectable (setup times). In other words, the travel time makes up for the major influencing factor of the order processing time. Furthermore, assuming that the order pickers travel at a constant speed, minimization of the total travel time is equivalent to (and can be achieved by) the minimization of the total length of all picker tours necessary to collect all items of a given set of customer orders (Jarvis & McDowell, 1991).

3. Order Batching

3.1 Problem Definition and Analysis

As has been mentioned before, the required items are collected on tours through the warehouse, where the number of stops on each tour is limited by the available capacity of the picking vehicle / picking device on the one hand and by the capacity requirements of the items to be picked on the other. Customer orders can be combined (*batched*) into to picking orders (*batches*) until the capacity of the device is exhausted. For the definition of the capacity of the picking device various criteria are used in the literature, e.g. the capacity can be expressed in number of customer orders (Le-Duc & de Koster, 2007), or in number of items (Bozer & Kile, 2008; Henn et al., 2010). Splitting of customer orders, i.e. the inclusion of items from the same customer order in several picking orders, is usually prohibited since it would result in an additional, not-acceptable sorting effort.

Based on these conclusions, the (static) order batching problem can be defined as follows: How, given routing strategy and capacity of the picking device, can a given set of customer orders with known storage locations be grouped (batched) into picking orders such that the total length of all picker tours is minimized? (Wäscher, 2004).

For a simple example, namely the case of two customer orders and S-Shape routing, Fig. 5 demonstrates that benefits may arise, indeed, from collecting the items requested by two customer orders on a single tour instead of collecting them on two separate tours. Figs. 5a and 5b depict the two tours based on separate picking, while Fig. 5c illustrates the single tour resulting from the batching of both customer orders. The length of the resulting tour is obviously shorter than the total length of the two separate tours. As can easily be concluded further, the benefits will be particularly large whenever the items of the batched customer orders have nearby locations.

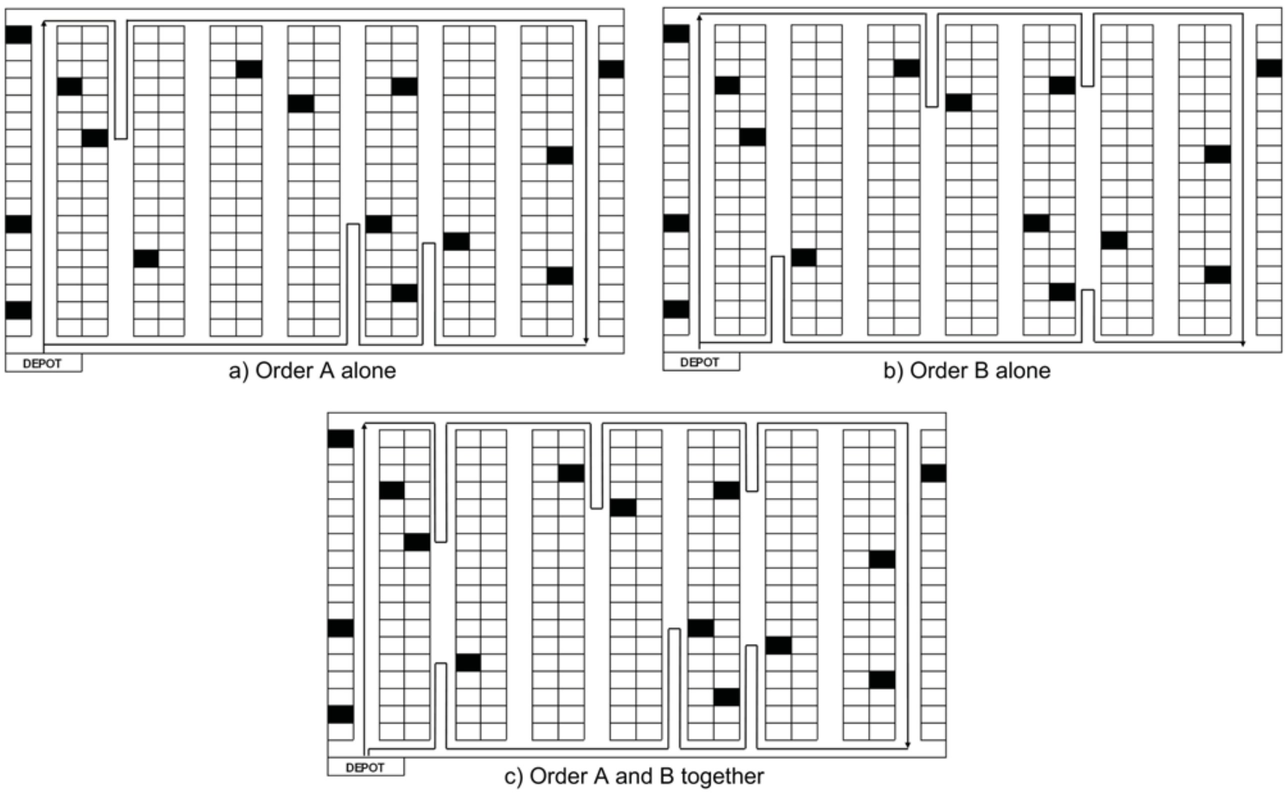

Fig. 5. Benefits arising from the combination of two customer orders

The order batching problem as described above is known to be NP-hard (in the strong sense) if the number of orders per batch is larger than two. This result can be proven by showing that the partition-into-triangles problem (Garey & Johnson, 1979) is polynomial reducible to the order batching problem (Gademann & van de Velde, 2005).

The order batching problem features some similarities with the capacitated vehicle routing problem, however differs from that with respect to the (customer) order integrity condition, i.e. all items of a customer order must be picked on the same tour. Thus, traditional solution approaches to the capacitated vehicle routing problem cannot be applied directly to the order batching problem (Bozer & Kile, 2008).

3.2 Model Formulation

A straightforward optimization model of the order batching problem has been introduced by Gademann & van de Velde (2005). In this model, all feasible batches (i.e. all batches which do not exceed the capacity of the picking device) are considered explicitly. For the presentation of the model, the following parameters are introduced:

J	set of customer orders, where $J = \{1,...,n\}$;
C	capacity of the picking device;
c_j	capacity utilization required by customer order j ($j \in J$);
I	set of all feasible batches;
d_i	length of a picking tour in which all orders of batch i ($i \in I$) are collected.

Let $a_i = (a_{i1},\ldots,a_{in})$ be a vector of binary entries a_{ij}, stating whether customer order j ($j \in J$) is included in batch i ($a_{ij} = 1$) or not ($a_{ij} = 0$). Then the set of feasible batches is characterized by the fact that the capacity constraint is not violated, therefore

$$\sum_{j \in J} c_j a_{ij} \le C, \qquad \forall i \in I \tag{1}$$

holds. Furthermore, the following decision variables are used:

x_i binary decision variable, $x_i = 1$, if batch i ($i \in I$) is chosen, or $x_i = 0$, otherwise.

The optimization model can then be formulated as follows:

$$\min \sum_{i \in I} d_i x_i \tag{2}$$

subject to

$$\sum_{i \in I} a_{ij} x_i = 1, \qquad \forall j \in J; \tag{3}$$

$$x_i \in \{0,1\}, \qquad \forall i \in I. \tag{4}$$

The sets of constraints (3) and (4) ensure that a set of batches is chosen in such a way that each customer order is included in exactly one of these batches.

Another optimization model for order batching in a single block layout has been introduced by Bozer & Kile (2008). However, the model is only applicable for the special case of S-Shape routing in narrow aisles where reversing is not permitted. With respect to these special conditions we do not discuss any further details here.

3.3 Solution Approaches

It is important to note that the number of possible batches and, consequently, the number of binary decision variables in the optimization system (1)-(4) grows exponentially with the number of customer orders. Henn et al. (2010), e.g., report on problem instances consisting of 40 customer orders in which the number of feasible batches is larger than 350,000 for a warehouse with 900 storage locations. Thus, a solution approach which involves the application of commercial LP/IP solvers to an explicitly formulated optimization system (1)-(4) covers only a limited range of problem instances. Henn et al. (2010) were only able to solve problem instances with at most 50 customer orders to optimality. For a large number of instances, the LP/IP solver was only able to generate feasible solutions but was unable to prove their optimality, since memory restrictions of the used PC were violated.

Based on the model formulation (1)-(4), Gademann & van de Velde (2005) present a column generation approach which starts from an initial set of batches and consecutively adds new batches as long as they improve the solution. The initial set of batches is determined by an iterated descent algorithm. Then – limited to these batches – the linear relaxation of the optimization system (1)-(4) is solved. In each iteration, a pricing algorithm generates one or several batches whose reduced costs are minimal. These batches are added to the (relaxed) model and the model is solved again. These steps are repeated until no further batch can be identified which would improve the current solution. If the obtained solution is not integral, a branch-and-price algorithm is used in order to generate an optimal integral solution for the order batching problem.

Gademann & van de Velde (2005) carried out an extensive numerical study on their exact solution approach. In their experiments they focus on a warehouse with 400 storage locations. They investigate several problem classes with up to 30 customer orders and 10 customer orders as a maximal capacity of the picking device. The algorithm is able to provide optimal solutions for almost all considered instances within a few minutes. The computing times increase significantly with an increasing number of customer orders; therefore the algorithm is not applicable for larger problem instances.

Due to the fact that the existing exact solution approaches to the order batching problem have only a limited applicability, real-world problems will have to be solved by means of heuristic approaches. These heuristic approaches can be classified into two groups, namely constructive solution approaches and metaheuristics.

4. Constructive Solution Approaches

Constructive solution approaches can be distinguished into priority rule-based algorithms, seed algorithms, savings algorithms and data mining approaches.

4.1 Priority Rule-Based Algorithms

Priority rule-based algorithms consist of a two-step procedure: In the first step, priorities are assigned to the customer orders. In the second step, in accordance with the previously assigned priorities, these customer orders are assigned successively to batches ensuring that the capacity constraint is not violated. A pseudocode for this type of algorithm is presented in Fig. 6.

Several rules have been suggested in the literature for the determination of priorities. The probably most straightforward one consists of assigning the priorities to the customer orders as they come in, i.e. according to the First-Come-First-Served (FCFS) rule. We note that this will practically result in a random sequence of the customer orders. Thus one cannot really expect that good-quality solutions will be provided by a procedure of this kind. However, the results from the application of the FCFS heuristic are commonly used as benchmarks in numerical experiments with other solution algorithms.

Gibson & Sharp (1992) introduced the application of two-dimensional and four-dimensional space-filling curves, Pan & Liu (1995) suggested the application of six-dimensional space-filling curves for the determination of priorities. The coordinates of the storage locations of the demanded items in a customer order are mapped onto a theta-value on the unit circle. This value defines the priority of the customer order. The order with the largest value receives the highest priority. Ruben & Jacobs (1999) propose that the customer orders are sorted according to a so-called order envelope. The envelop of a customer order is defined as the pair of aisle numbers where the first number corresponds to the leftmost and the second number corresponds to the rightmost aisle from which an item has to be picked.

Customer orders may be assigned to batches either sequentially according to the Next-Fit Rule, or simultaneously according to the First-Fit or the Best-Fit Rule. As for the Next-Fit Rule, customer orders are added to a batch until the capacity limit of the picking device is reached; in this case a new batch is opened. According to the First-Fit Rule batches are numbered in the order in which they were started (opened); the next customer order is allocated to a batch which possesses the smallest number and still provides sufficient capacity for the accommodation of the customer order (Ruben & Jacobs 1999). The Best-Fit Rule assigns an order to the batch with the least remaining capacity (Wäscher, 2004).

```
assign a priority to each customer order;
sort all orders according to non-ascending priorities;
open the first batch and assign the first order to it;
while there exist unassigned orders do
   select next order;
   if order can be assigned to an open batch then
     select open batch;
     assign order to this batch;
   else
     open new batch;
     assign order to this batch;
   endif
endwhile
```

Fig. 6. General principle of priority rule-based algorithms

4.2 Seed Algorithms

Seed algorithms, introduced by Elsayed (1981), generate batches sequentially by means of a two-phase procedure: a seed selection phase and an order congruency phase. During the seed selection phase, an initial order ("seed") is chosen for a batch which has just been opened. A large variety of rules is available for the selection of the seed (for some examples see Table 1). Furthermore, the seed can be determined in a *single mode* (where only the first order in the batch defines the seed) or in *cumulative mode* (where all orders included in the batch define the seed). Afterwards, in the order congruency phase, unassigned customer orders are added to the seed according to an order-congruency rule (for some examples see Table 2), which measures the "distance" from a customer order not yet allocated to the seed of the batch. A pseudocode for seed algorithms is depicted in Fig. 7.

Name of the Rule	**From the set of unassigned customer orders select one order...**
Random Seed (Gibson & Sharp, 1992)	... randomly.
Smallest (Largest) Number of Items (Elsayed & Stern, 1983)	... which consists of the smallest (largest) number of items.
Smallest (Largest) Number of Picking Locations (Elsayed & Stern, 1983; Elsayed, 1981)	... which possesses the smallest (largest) number of locations the order picker will have to visit in order to collect the items of this order.
Smallest (Largest) Number of Picking Aisles (Ho & Tseng, 2006; de Koster et al., 1999b)	... which possesses the smallest (largest) number of aisles the order picker will have to enter in order to collect the items of this order.
Smallest (Greatest) Location-Aisle Ratio (Ho & Tseng, 2006)	... with the smallest (largest) ratio of the number of storage locations to be visited over the number of aisles to be entered.
Smallest (Greatest) Aisle-Simple-Weight Sum (Ho et al., 2008)	... for which the sum of the weights of the aisles to be entered is minimal (maximal). The weight of an aisle is equal to its index, i.e. the weight of an aisle increases with its distance from the depot.
Smallest Aisle-Exponential-Weight Sum (Ho & Tseng, 2006)	... for which the sum of the weights of the aisles to be entered is minimal. The weight of an aisle depends exponentially on the index of the aisle.
Smallest (Greatest) Rectangular-Covering Area (Ho et al., 2008)	... for which the smallest rectangle is minimal (maximal) by which the storage locations of all required items of this order can be covered.
Shortest Average Rectangular (Euclidean) Distance to the Depot (Ho et al., 2008)	... for which the average rectilinear (Euclidean) distance between the depot and the storage locations to be visited is minimal.
Shortest Average Aisle Distance to the Depot (Ho et al., 2008)	... for which the average distance along the cross aisle between the depot and the storage locations to be visited is minimal.
Farthest Storage Location (de Koster et al., 1999b)	... which requires the collection of an item located farthest from the depot.
Longest Travel Time (de Koster et al., 1999b)	... which requires the longest travel time.
Largest Aisle Range (de Koster et al., 1999b)	... with the largest aisle range, i.e. the absolute difference between the number of the leftmost aisle and the number of the rightmost aisle to be entered.

Table 1. Examples of seed selection rules

Name of the Rule	**Add an order…**
Smallest Number of Additional Picking Locations (Aisles) (Rosenwein, 1996; Ho & Tseng, 2006)	… such that the number of additional picking locations (aisles) to be visited is minimal.
Smallest (Greatest) Overlapping Covering Area (Ho et al., 2008)	... such that the overlapping area between the smallest rectangle covering the storage locations of the items in this order (order covering rectangle) and the smallest rectangle covering the locations of the items already in the batch (batch covering rectangle) is minimal (maximal).
Smallest (Greatest) Additional Covering Area (Ho et al., 2008)	... such that the difference in size between the original batch covering rectangle and the new batch covering rectangle (i.e. the one also including the additional order) is minimal (maximal).
Shortest Average Mutual-Nearest-Rectangular (Euclidean) Distance (Ho et al., 2008)	... which minimizes the minimal average mutual-nearest-rectangular (Euclidean) distance. This distance is determined as follows: At first, the sum of the rectangular (Euclidean) distances from each pick location of the order to the respective closest pick locations of the batch is divided by the number of items in the order. Secondly, the sum of the rectangular (Euclidean) distances from each pick location of the batch to the respective closest pick location of the order is divided by the number of items in the batch. The mean of these two values gives the average mutual-nearest-rectangular (Euclidean) distance.
Shortest Average Mutual-nearest-Aisle Distance (Ho et al., 2008)	... which minimizes the average mutual-nearest-aisle distance. This distance can be determined similar to the average mutual-nearest-rectangular distance, whereas the distance between two pick locations is defined according to the distance on the cross aisle between the two pick locations.
Smallest Sum I /II (Gibson & Sharp, 1992; Pan & Liu, 1995)	… for which the sum of the distances between the storage locations of the batch (order) and the closest item of the order (batch) is minimal.
Smallest Center of Gravity (Rosenwein, 1996)	… for which the absolute difference between the centre of gravity of the order and the centre of gravity of the batch is minimal. The centre of gravity is defined as the average number of aisles which have to be entered by the order picker.
Time Saving (de Koster et al., 1999b)	… for which the travel time reduction is maximal when being added to the batch, i.e. when batch and order are picked on one tour instead of two separate tours.
Greatest Number of Identical Picking Locations (Aisles)	… with has the largest number of identical pick locations (aisles) in common with the batch.

(Elsayed & Stern, 1983; Ho & Tseng, 2006)	
Greatest Picking Location (Aisles) Similarity Ratio (Ho & Tseng, 2006)	… which maximize the ratio of the number of picking locations (aisles) batch and order have in common over the number of picking locations (aisles) of batch plus order.
Greatest Picking Location (Aisles) Covering Ratio (Ho & Tseng, 2006)	… which maximizes the ratio of the number of picking locations (aisles) batch and order have in common over the number of picking locations (aisles) of the order.

Table 2. Examples of order-congruency rules

```
while there exist unassigned customer orders do
  open a new batch;
  select an unassigned order as seed order;
  while orders exist which does not exceed the
    remaining capacity of the current batch then
    select an unassigned order;
    assign order to open batch;
  endwhile
  close current batch;
endwhile
```

Fig. 7. General principle of seed algorithms

4.3 Savings Algorithms

Savings algorithms are based on the Clarke-and-Wright Algorithm for the vehicle routing problem (Clarke & Wright, 1964) which has been adapted in several ways for the order batching problem. In the initial version (C&W(i)) of the algorithm for the order batching problem, for each combination of customer orders savings are computed which can be obtained in terms of tour length reduction by assigning the items of the customer orders to one (large) batch instead of collecting them separately. Starting with the pair of orders that provides the highest savings, the pairs are considered one after another in a non-ascending order and checked with respect to the following three situations: (1) None of the two orders has been assigned to a batch yet; in this case a new batch will be opened and the orders will be assigned to it. (2) One of the orders has already been assigned to a batch; the other one will then added to the batch if the remaining capacity is sufficient; otherwise the next pair of orders will be considered. (3) Both orders have already been assigned; then the next pair of orders is considered. All orders which are left unallocated at the end of the process will be assigned to an individual batch each (Elsayed & Unal, 1989).

In the second variant of the algorithm, denoted by C&W(ii), savings are recalculated each time a new assignment of customer orders to batches has been made. The (elementary) orders which have already been combined into a batch are excluded from these calculations, in which, on the other hand, the new batch is treated as a new "large" order (Elsayed & Unal, 1989). A pseudocode of this algorithm is presented in Fig. 8. Referring to C&W(ii),

Bozer & Kile (2008) suggest a normalized savings algorithm in which normalized instead of absolute savings are used, i.e. the travel time which can be saved by picking two orders on the same tour is divided by the time necessary if both orders are picked on separated tours.

A drawback of C&W(i) can be seen in the fact that the algorithm may generate solutions with a large number of batches. In order to control the number of batches, C&W(i) can be modified in the following way: The initial savings matrix is modified each time customer orders have been assigned, i.e. for those pairs of customer orders of which both orders have not yet been assigned to a batch, the savings are reduced by a constant value. Therefore, the algorithm tends to select pairs of customer orders where at least one order is already assigned to a batch (C&W(iii); Elsayed & Unal, 1989).

The EQUAL (Elsayed & Unal, 1989) algorithm generates batches sequentially and uses the seed algorithm principle. The pair of customer orders with the highest saving is allocated to a batch and considered as an initial seed. Then a single order is assigned to the batch which does not violate the capacity constraint and results in the highest savings in combination with the seed. The seed and the assigned order form the new seed. If none of the remaining unassigned orders fits into the batch, then a new batch is opened.

```
repeat
   calculate savings $sav_{ii'}$ for all pairs (i,i') of orders
      i and i';
   repeat
      choose the pair with highest savings;
      if combination of both orders does not violate the capacity
            of the picking device then
         combine both orders to a new larger order;
         remove both orders from the list;
         include the new larger order in the list;
      else
         set savings to zero;
      endif
   until two orders are combined or all pairs have been
      considered;
until no pair was combined in the last iteration;
each of the remaining orders serves as a single batch;
```

Fig. 8. Savings algorithm C&W(ii)

In the Small-and-Large Algorithm (Elsayed & Unal, 1989) two subsets are defined, namely a set of large customer orders whose number of requested items exceeds a predefined number and a set of small ones made up by the remaining ones. The set of large orders is assigned to batches by application of the EQUAL algorithm. The small customer orders are sorted in a decreasing order of their size; then – in the sequence given by this sorting – each order is allocated to a batch where it does not violate the capacity constraint and generates the highest savings. If during the procedure a “small” order cannot be added to any of the existing batches, this order is assigned to a new batch. This batch is included in the set of batches and considered in the following iterations.

4.4. Data Mining Approach

Finally, Chen & Wu (2005) describe an order batching approach based on data mining and integer programming. In this approach, at first similarities of customer orders are determined by means of an association rule. For each pair of orders a support value (or order correlation measure) is obtained. In a 0-1 integer programming approach orders are clustered into batches such that the sum of all support values to the batch medians, an order which serves as basis for each batch, is maximized.

5. Metaheuristics

5.1. Local Search

The general principle of local search-based heuristics consists of exploring the neighbourhood of a solution in order to identify a new solution with a smaller objective function value. For a solution S a solution is called neighbour solution if it can be obtained by applying a single local transformation (”move”) to S. Classic local search generates a sequence of solutions S_0, S_1, S_2, . . . , where each member of the sequence is a neighbour of its predecessor. Each element possesses a smaller objective function value than the previous one. Classic local search stops at a local minimum, i.e. when no neighbour solution can be found that has a smaller objective function value than the incumbent solution. Classic local search suffers from the fact that it gets stuck in a local minimum which is probably far from the global minimum. Therefore, the local search principle has been modified in several ways in order to overcome such local minima.

The first local search-based approach for the Order Batching Problem was suggested by Gademann & van de Velde (2005). Their method starts from an initial solution generated by means of the FCFS rule. Neighbour solutions are obtained by so-called SWAP moves, in which two orders from different batches are interchanged. Since Gademann & van de Velde (2005) consider a problem in which the capacity of the picking device is defined with respect to the number of customer orders, the algorithm operates only on feasible neighbour solutions. The authors have implemented a first-improvement strategy, i.e. the first solution from the neighbourhood which has been identified to possess a better (smaller) objective function value is accepted and taken as the new incumbent solution. When a local minimum has been reached, the solution is perturbed by a sequence of three operations. In each operation three customer orders from three different batches are interchanged at random. The solution obtained from these perturbations is taken as a new incumbent solution from which the improvement phase is started again. This series of operations is repeated for a predefined number of iterations.

Henn et al. (2010) propose the application of Iterated Local Search (ILS) to the order batching problem. ILS tries to intensify the search for improved solutions in the vicinity of local minima. It consists of two phases, a perturbation and a local search phase. In the perturbation phase, the incumbent solution is partially modified (perturbed). In the local search phase, proceeding from this solution, one tries to identify an improved solution. The solution stemming from the local search phase has to pass an acceptance criterion in order to become the new incumbent solution; otherwise the previous solution remains the incumbent solution to which another perturbation is applied. The two phases are repeated in turn until a termination condition is met.

With respect to the order batching problem, Henn et al. (2010) suggest the following: An initial solution is generated by means of the FCFS rule. The local search phase includes SWAP moves (as defined above) and SHIFT moves (in which one customer order is selected and assigned to a different batch). One starts with a series of SWAP moves, until no further improvement can be obtained by this type of moves. Then one switches to a series of SHIFT moves, again until no further improvement can be achieved. Then one goes back to a series of SWAP moves etc. This sequence of alternating stages of SWAP and SHIFT moves is repeated until no further improvement can be obtained. In the perturbation phase two different batches are selected at random and a randomly generated number of orders from the first batch are moved to the second one, and vice versa. Orders whose addition to a batch would result in a violation of the capacity constraint will be assigned to a new batch. A new solution is accepted as an incumbent solution if its objective function value is lower than the one of the currently best known solution. Furthermore, a few deteriorating steps are allowed if a sequence of perturbation phases and local search phases applied to a particular incumbent solution does not lead to a new global best solution within a certain time limit.

Albareda-Sambola et al. (2009) apply Variable Neighbourhood Search (VNS) to the order batching problem. They define three different kinds of neighbourhoods, based on the following moves: (i) Assignment of one order to a different batch (SHIFT move). (ii) Assignment of up to two orders from one batch to other batches; this includes a simple SHIFT move, the assignment of two orders from one batch to another batch, and the assignment of two orders of one batch to two different batches. (iii) Assignment of up to two orders from one or two batches to one or two other batches; this set includes the moves from neighbourhoods (i) and (ii), plus the SWAP move, the assignment of two orders from different batches to one single batch, and the transfer from one order to a another batch while an order from that batch is transferred to a third batch. In all three cases, only moves to feasible solutions are accepted. Starting from neighbourhood (i), the algorithm explores the three neighbourhoods successively. Whenever no improvement can be identified within one neighbourhood, it proceeds to the next (more extensively-defined) one. Having accepted an improved solution within neighbourhood (iii), the algorithm returns to exploring neighbourhood (i), etc. The algorithm terminates if the incumbent solution is optimal for all three neighbourhoods.

5.2. Tabu Search

Tabu Search - developed by Glover (1986) - aims at simulating human memory processes by means of a so-called tabu list. This list records moves applied in previous iterations. In order to avoid cycling and to diversify the search, the application of these moves is set forbidden (″tabu″) for a particular number of iterations. In each iteration Tabu Search considers only those elements of the neighbourhood which can be obtained by a non-tabu move. From this solution the one with smallest objective function value, which may not be smaller than the objective function value of the incumbent solution, is chosen as the next incumbent solution.

Henn & Wäscher (2010) explore several variants of Tabu Search for the order batching problem. Options for the generation of initial solutions include the FCFS-rule and C&W(ii).

Three neighbourhoods are investigated, namely those based on (1) SWAP moves, (2) SHIFT moves, and (3) SWAP or SHIFT moves. The neighbourhoods may either be explored completely or partially, only.

Henn & Wäscher (2010) also introduce the Attribute-based Hill Climber (ABHC) heuristic for the order batching problem. ABHC is an almost parameter-free heuristic based on a simple Tabu Search principle which can be described as follows: For each problem, a set of attributes is introduced. An attribute can be any specific solution feature. During the local search phase a solution can be accepted if and only if it possesses the smallest objective function value found so far for at least one attribute. The algorithm stops if the current neighbourhood contains no solution that represents a best solution for at least one attribute. The advantage of ABHC can be seen in the fact that – related to the design of the algorithm – only three decisions have to be taken, namely with respect to the choice of the initial solution, the neighbourhood structure, and the set of attributes (Whittley & Smith, 2004).

For the determination of the initial solution and for the neighbourhood structure Henn & Wäscher (2010) make use of the same options as for Tabu Search. As for the attributes, two sets are proposed: The first attribute set characterizes each solution by pairs of customer orders which are assigned to the same batch. The second set of attributes is related to the assignment of customer orders to batches.

5.3. Population-based Approaches

The Rank-based Ant System (RBAS) is a population-based solution approach in which each ant presents a single solution (Bullnheimer et al, 1999). Henn et al. (2010) modify the RBAS for the order batching problem. In their method a specified number of ants is observed during a period of several iterations. For each ant the algorithm starts from a solution in which each order forms a single batch. In the subsequent steps, the batches are combined as long as the capacity constraint for the picking device is not violated. For each possible combination of two batches into one, the savings (cf. Section 4.3) and the pheromone intensity are computed. The pheromone intensity of a batch combination is determined as a relative impression, namely as the sum of the pheromones of all order combinations (i.e. of all pairs of customer orders, where one order is in the first batch and one order is in the second batch) over the number of possible order combinations between the two batches. Savings and pheromone intensity define the probability according to which two batches will be combined. If no further feasible combinations of batches can be identified, an attempt is made to improve the obtained solution by applying an elementary local search function. The process is repeated for each ant that is used. After the last ant of an iteration has been taken care of, the pheromones for all order combinations are updated: in general, a fraction of the pheromone “evaporates”, while the “good” solutions receive an additional amount of pheromone.

Genetic algorithms iteratively generate a large number of possible solutions and select the best ones. These solutions are modified (mutation) and combined (cross-over) in order to generate new solutions. Hsu et al. (2005) have modified this general approach for the order batching problem. Each solution is represented by a string of integers, which groups each customer order into a particular batch. The fitness of a solution is calculated as the difference between the length of the longest tour in the population and the tour length of this solution. The authors also propose several crossover and mutation mechanisms for their algorithm.

6. Performance of Heuristic Algorithms

An up-to-date study which considers a comprehensive set of problem parameters and provides an in-depth analysis of the performance of a representative set of the solution methods for the order batching problem does not exist in the literature so far (Gu et al., 2007). Published results of numerical experiments are usually based on very diverse settings (e.g. size and dimensions of the warehouse, number and size of customer orders, capacity of the picking device, demand structure, or the used routing policy). Thus, general conclusions, e.g. concerning the superiority of one method over another, are hardly possible at this stage.
Gibson & Sharp (1992) analyze the performance of priority rule-based algorithms using space-filling curves, and that of a basic seed algorithm which combines *Random Seed* as a seed selection rule and *Smallest Sum I* as an order congruency rule. Their numerical experiments are based on a fixed warehouse with 800 storage locations. It is shown that the seed algorithm in which distances between locations are measured according to an aisle metric provides the shortest tour lengths.

Rosenwein (1996) analyzes the performance of the following order congruency rules: *Smallest Center of Gravity*, *Smallest Sum I*, and *Smallest Number of Additional Picking Aisles*. For a warehouse with 750 storage locations, the *Smallest Number of Additional Picking Aisles* generates smaller average tour lengths than the two other approaches.

De Koster et al. (1999b) have carried out extensive numerical experiments in order to investigate the performance of seed and savings algorithms. In their experiments the heuristics are evaluated with respect to parameters like the size of the warehouse (240, 400, 1250 storage locations), the demand structure (uniformly, class-based), and the number of orders per batch. Their experiments show that for seed algorithms the cumulative mode outperforms the single mode. Furthermore, *Largest Number of Picking Aisles*, *Longest Travel Time,* and *Largest Aisle Range* are the seed selection rules by which the shortest total tour lengths are obtained. In comparison to FCFS, the best seed-algorithms improve the tour lengths by 19.7% (small warehouse size) to 7.5% (large warehouse size).

Ho & Tseng (2006) and Ho et al. (2008), in two comprehensive numerical studies, analyze the performance of seed algorithms. They compare various combinations of seed selection and order congruency rules to each other. In both articles an identical test setting for a warehouse with 384 storage locations is used. In the experiments of Ho & Tseng (2006) a combination of *Smallest Number of Picking Aisles* (seed selection rule) and *Smallest Number of Additional Picking Aisles* (order-congruency rule) gives the smallest tour lengths. Ho et al. (2008) observed that these results can be improved by combination of *Smallest Number of Picking Aisles* and *Shortest Average Mutual-nearest-Aisle Distance*.

In their analysis of savings algorithms, de Koster et al. (1999b) show that C&W(ii) generates the smallest tour lengths among the described variants. For a small warehouse, the tour lengths obtained by application of C&W(ii) are more than 20% shorter than the ones obtained by FCFS.

In de Koster et al. (1999b), also savings and seed algorithms are compared. For a small warehouse, C&W(ii) provides smaller tour lengths than the best seed algorithm, whereas for a larger warehouse the best seed algorithm outperforms C&W(ii) in terms of solution quality. However, C&W(ii) consumes about ten times the computing time of the other savings algorithms and about 100-200 times the computing time required for the seed algorithms.

Bozer & Kile (2008) compare the lower and upper bound from their model (cf. Section 3.2) to the results obtained by their Normalized Savings approach. In total, results obtained by the Normalized Savings approach are 11% above the lower bound and 6 - 7.5% above the upper bound. In their experiments the number of customer orders is limited to 25.

In the numerical experiments of Chen & Wu (2005) different warehouse sizes (ranging from 40 up to 300 storage locations) are considered. The authors demonstrate that their approach which combines data mining and integer programming may reduce the tour lengths provided by FCFS significantly.

Henn et al. (2010) benchmark their versions of ILS and RBAS against FCFS and C&W(ii). Extensive numerical experiments have been carried out in which a warehouse with 900 storage locations and a class-based demand structure have been assumed. In these experiments the picker routing problem was solved by means of the S-Shape and Largest Gap Heuristics. The authors find that – in comparison to FCFS - C&W(ii) reduces the total tour length by 17%, while ILS and RBAS improve the results by approximately 20%. The results from ILS on one hand and RBAS on the other differ for less than 1%.

For a similar test setting, Henn & Wäscher (2010) demonstrate that the best performing Tabu Search and ABHC variants manage to improve the solutions obtained by C&W(ii) by 4.1% and 4.6%, respectively. The results provided by the iterated descent algorithm of Gademann & van de Velde (2005) differ from the optimal total tour length by a 1% on average and up to 6 % as the maximum.

Albareda-Sambola et al. (2009) perform numerical experiments of their VNS approach using the warehouse layouts described in de Koster et al. (1999b) and Ho & Tseng (2006). In their experiments they benchmark VNS against FCFS, and several seed and savings approaches. It can be concluded that among these heuristics VNS was able to find the best solutions in most cases and improves the results obtained by FCFS by 19% on average.

Hsu et al. (2005) benchmark their genetic algorithm against the results obtained by FCFS, considering warehouse sizes differing between 40 and 400 locations. They obtain improvements of up to 31%.

As stated above, it is difficult to compare the presented results with each other due to the different parameter settings used in the numerical experiments. In order to make results more comparable, it would be useful for future research to design experiments similar to already existing ones. Furthermore, a systematic and comprehensive study is desirable in order to evaluate the performance of existing solution approaches.

7. Wave Picking

Wave picking is a variant of order picking in which a large set of customer orders (called wave) for a joint destination is released simultaneously. Different to the approaches described before, the workload can be spread over several order pickers. Wave picking aims at collecting the items of the set of orders as fast as possible, even if this would result in a non-minimal tour length. Typically, this situation arises when this set of orders belongs to the same truck load and the truck cannot leave until all orders are completed. The next wave can only be started when the previous one is completed (Gademann et al., 2001).

In order to control the picking and shipping process, the following optimization problem can be stated: How, given routing strategy, capacity of the picking device and a fixed number of order pickers, can a given set of customer orders with known storage locations be grouped (batched) into picking orders such that the maximal processing time of all batches is minimized? It is usually assumed that the order pickers do not block each other when entering the aisles or travelling through them. Thus, the impact of congestions is not explicitly considered in the corresponding planning models and solution approaches.

A similar model to the one described in Section 3.2 can also be used for a representation of this problem. The following additional notation is used:

pt_i	processing time of batch i ($i \in I$), i.e. the time which is necessary to collect all customer orders of batch i;
m’	number of order pickers, i.e. the number of batches which can be processed in parallel.

$$\min \max_{i \in I} \; pt_i x_i \tag{5}$$

subject to

$$\sum_{i \in I} a_{ij} x_i = 1, \qquad \forall j \in J; \tag{6}$$

$$\sum_{i \in I} x_i \le m'; \tag{7}$$

$$x_i \in \{0,1\}, \qquad \forall i \in I. \tag{8}$$

Again, constraints (6) and (8) ensure that a set of batches is chosen in such a way that each customer order is included in exactly one of the chosen batches. (7) guarantees that at most m’ batches are selected. The objective function (5) minimizes the maximal processing time of the batches. Bozer & Kile (2008) point out that solutions of this optimization problem may include situations in which order pickers may be idle while remaining orders are still being collected. By reducing the partition problem (Garey & Johnson, 1979) to the wave picking problem, Gademann et al. (2001) show that the corresponding decision variant is *NP*-complete.

For the solution of the above-stated problem Gademann et al. (2001) propose a branch-and-bound algorithm and a 2-opt heuristic. They evaluate their algorithms for several problem classes. For problem classes with up to 24 customer orders their branch-and-bound algorithm was able to solve all instances. For larger problem classes only a subset of instances could be solved within a time limit of 600 seconds. From extensive numerical experiments the authors conclude that the number of order pickers and the number of customer orders have a significant impact on the computing time of the algorithm. They also demonstrate that maximal processing times provided by the 2-opt heuristic are very close to the optimal ones.

8. Order Batching and Batch Scheduling

In order picking systems it is not uncommon that the customer orders have to be completed and provided by certain due dates. In distribution warehouses due dates have to be met in order to guarantee the scheduled departure of trucks (Gademann et al., 2001). In material warehouses which provide the input to a production system (internal customers), on-time retrievals from the warehouse are vital in order to avoid production delays. In a Just-in-Time environment, in addition to not allowing orders to be late, it is also not acceptable that the items are provided a long time ahead of the due date, since that would result in an unnecessary accumulation of material or work-in-progress. In such cases, instead of measuring the quality of a solution by means of the total picking time or the total length of the picking tours, the batching of customer orders into picking orders will have to be evaluated with respect to both earliness and tardiness of the orders. The weighted sum of the (total) earliness and the (total) tardiness of all customer orders may be minimized in order to model these aspects (Elsayed et al., 1993). The composition of the batches, but also the sequence according to which the batches are processed and the corresponding *release times* (i.e. the points in time when the various batches and/or customer orders are started) determine whether and how this goal is met.

With respect to the described situation, the following problem can be stated: How, given routing strategy and capacity of the picking device, can a given set of customer orders with known due dates and storage locations be grouped (batched) into picking orders such that the weighted sum of the total earliness and the total tardiness of all customer orders is minimized?

Elsayed & Lee (1996) have suggested an optimization model for order batching for an AS/RS which minimizes the total tardiness. We adopt this model here for the above described, more general goal and introduce the following index sets and constants in addition to those which have already been used in the above-described models:

dd_j	due date of customer order j ($j \in J$);
J_i	set of customer orders which are included in batch i ($i \in I$), i.e. $J_i=\{j \in J \mid a_{ij}=1\}$;
M	a sufficiently large (positive) number;
α	(relative) weight for the total earliness of all customer orders;
β	(relative) weight for the total tardiness of all customer orders.

Furthermore, the following variables are introduced:

ct_i	completion time of batch i ($i \in I$); point in time when the order picker returns to the depot after having collected all items of batch i;
v_{ik}	binary decision variable which describes whether batch i is released directly before batch k ($v_{ik} = 1$) or not ($v_{ik} = 0$) ($i, k \in I$);
ta_{ij}	tardiness of customer order j ($j \in J$) in batch i ($i \in I$);
ea_{ij}	earliness of customer order j ($j \in J$) in batch i ($i \in I$).

Then the following mixed integer programming model can be formulated:

$$\min \alpha \sum_{i \in I} \sum_{j \in J} a_{ij} ea_{ij} + \beta \sum_{i \in I} \sum_{j \in J} a_{ij} ta_{ij} \tag{9}$$

subject to:

$$\sum_{i \in I} a_{ij} x_j = 1, \ \forall j \in J; \tag{10}$$

$$pt_i x_i \le ct_i, \ \forall i \in I; \tag{11}$$

$$ct_i - ct_k + Mv_{ik} \ge pt_j x_j, \ \forall i,k \in I \text{ and } i < k; \tag{12}$$

$$ct_k - ct_i + M(1 - v_{ik}) \ge pt_k x_k, \ \forall i,k \in I \text{ and } i < k; \tag{13}$$

$$ct_i - ta_{ij} \le dd_j x_i, \ \forall j \in J_i, \forall i \in I; \tag{14}$$

$$ea_{ij} - ct_i \leq dd_j x_i, \ \forall j \in J_i, \forall i \in I; \tag{15}$$

$$ct_i \geq 0, \ \forall i \in I; \tag{16}$$

$$ta_{ij} \geq 0, \ \forall i \in I, j \in J; \tag{17}$$

$$ea_{ij} \geq 0, \ \forall i \in I, j \in J; \tag{18}$$

$$x_i \in \{0,1\}, \ \forall i \in I; \tag{19}$$

$$v_{ik} \in \{0,1\}, \ \forall i,k \in I, i < k. \tag{20}$$

In this model, objective function (9) represents the weighted sum of the total earliness and the total tardiness of all customer orders. As in the previously-described models, it is guaranteed by constraints (10) and (19) that each customer order is assigned to exactly one batch. Inequalities (11) imply that the completion time of any batch is always greater than or equal to its processing time. For any pair (i, k) of batches i and k ($i,k \in I$) constraints (12) and (13) imply that the completion time of either batch i or batch k must be greater than or equal to the completion time of the immediately preceding batch plus the corresponding processing time. Constraints (14) determine the tardiness of each customer order i ($i \in I$); it is defined by the number of time units according to which the due date of i is exceeded by the completion time of the batch to which it has been assigned to. Analogously, constraints (15) determine the earliness of each customer order. Constraints (16), (17) and (18) ensure that the variables assume non-negative values which represent the completion times, the earliness and the tardiness of the customer orders. The variables v_{ik} which describe whether batch i is released directly before batch k ($i,k \in I$) are binary by definition; this is guaranteed by constraints (20).

Elsayed et al. (1993) suggest a solution method for the above problem which consists of three steps. In step 1, priorities are determined for the customer orders. Each customer order is considered as a single batch and its priority value is defined by the weighted sum of its due date and the corresponding processing time. Customer orders are ranked in ascending order of their priority index. Based on this sequence the objective function value is determined. This sequence is updated, if a pairwise exchange of two adjacent customer orders can be identified which would improve the objective function value. In step 2, customer orders are combined into batches according to the sequence established in step 1: For each customer order it is determined whether it is favourable to pick it separately or if it is better to add the customer order in one of the already existing batches. In the latter case the customer order is assigned to the first batch which results in a smaller objective function value. For each of the obtained batches, release times are determined (Step 3). Now an idle time may occur between the release time of a batch and the completion time of its predecessor. As a consequence the algorithm shifts batches along the time axes in order to obtain improved objective function values by reducing the total earliness, since earliness is penalized as well in the objective function.

Elsayed & Lee (1996) propose a solution approach for the generation and sequencing of batches when only the total tardiness has to be minimized. At first, the customer orders are sequenced according to their due dates and the times that would be needed if each customer order would be processed in a single batch. A first bound on the optimal total tardiness can be obtained by assuming that the items of each customer order would be collected on a single tour and sequencing the tours according to the order's position in the sequence. In order to improve this bound, three decision rules for the generation of batches are proposed and evaluated: (1) The Nearest Schedule Rule assigns the first customer order of the obtained sequence as a seed to the first batch. Those customer orders of the sequence are assigned to the batch if the inclusion does not increase the total tardiness and does not violate capacity restrictions. When no further customer order can be added, the first remaining customer order in the sequence will serve as seed order for the next batch. (2) The Shortest Service Time Rule adds orders to the seed which would possess the shortest processing time, if the customer order was processed in a separate tour. (3) In the Most Common Location Rule, a customer order is added to the seed which has the largest number of pick locations in common with the seed order. In numerical experiments it is shown that the Nearest Schedule Rule outperforms the two other rules and achieves results which are close to optimality. Additionally, the authors show how storage orders (orders where items have to be carried to the storage locations) can be included in the obtained sequence.

Tsai et al. (2008) consider an order batching and sequencing problem where the total travel costs (depending on the total travel time) have to be minimized and earliness and tardiness are penalized. Unlike in the approaches previously discussed, here the authors permit the splitting of orders; therefore, the items required by a single customer order may be collected on different tours. The batching problem is solved by means of a genetic algorithm. In this algorithm, a solution is represented by a sequence of integers. This sequence contains the indices of the batches to which the items requested in the customer order have been assigned. In order to determine the fitness of the generated solutions, a travelling salesman problem is solved by another genetic algorithm.

Won & Olafsson (2005) discuss a joint batching and picking problem where customer orders arrive at (known) different points in time. They formulate an optimization model where the objective function is a weighted sum of travel distance and the time period between the arrival and the time when the order picker starts to collect the items of a customer order. The authors observe a tradeoff between the necessary travel time and the length of the time period for which the customer order stays in the system. In order to solve the problem the authors propose a two-step heuristic. In this heuristic, batches are formed by application of the FCFS rule. The subsequent routing problem is then solved by a 2-opt procedure. Additionally, they present an algorithm which solves the batching and routing problem simultaneously.

9. Dynamic Order Batching

9.1 Problem Description

Whereas in the static case of order batching the characteristics of each customer order (i.e. the requested articles and the corresponding quantities) are known in advance, the dynamic case (also referred to as on-line batching; cf. Yu & de Koster, 2009) can be characterized as an order picking environment in which the customer orders arrive stochastically over time, and only when an order has arrived the information becomes available of which articles and respective quantities the order is composed of.

Under such conditions time window batching is prevalent, which can be carried out in two different ways, namely *variable time window batching* and *fixed time window batching* (Van Nieuwenhuyse & de Koster, 2009). In variable time window batching it is usually assumed that the capacity of the picking device is defined in the number of customer orders which can be accommodated. The order picker waits until a particular number of customer orders (smaller or equal to the capacity) has arrived and then collects the items of these orders on a single tour. In fixed time window batching all customer orders arriving during a particular time interval are assigned to batches. It has to be noted that all batching decisions, i.e. how the orders should be assigned to batches and how the batches should be released in time, are to be based on the known customer orders only.

The point in time when a customer order becomes available is called *arrival time* of the order. The *waiting time* of a customer order is defined by the length of the time interval between the arrival time of the order and its start time. The *turnover time* (also called *throughput time* or *response time*) of a customer order is equal to the length of the time period for which the order stays in the system, i.e. the time period between the arrival time of the customer order and its completion time.

The performance of a dynamic order picking system can be measured by the (average) turnover time of the customer orders. This measure can be seen as an indicator of the service level on the one hand, but also as an expression for the available capacity on the other, i.e. the number of orders which can be processed in a given period of time. Reduced turnover times result in improved service levels and increase the capacity of the warehouse.

Dynamic order batching deals with the following question: In a dynamic order picking environment, in which customer orders arrive over time, how should the customer orders (with given storage locations and given routing strategy) be grouped into picking orders such that the average turnover time of the customer orders is minimized?

9.2 Time Window Batching for the Minimization of Turnover Times

The time an order picker is expected to spend collecting the items of a batch is estimated by travel time models. These estimations depend on several parameters, e.g. the average number of items to be picked on a tour, the demand distribution, or the layout of the warehouse and its dimensions (Caron et al., 2000).

In the literature, the order picking system is regarded as a continuous system with an infinite number of arrivals of customer orders. Chew & Tang (1999) present a travel time model of a single block order picking warehouse with variable time window batching. Based on this model, an estimation for the determination of the number of customer orders is developed which should be assigned to a single batch in order to minimize the average turnover time of a customer order. On the basis of S-Shape routing, they carry out a theoretical analysis of travel and processing times for the first customer order in a batch. The system is designed as a queuing network with two queues. In the first one, customer orders arrive according to a Poisson-process and batches are generated by means of the FCFS rule. If a particular number of customer orders is in the first queue, these customer orders are assigned to a batch and move to the second queue. The batches in the second queue are released successively according to the availability of order pickers. In numerical experiments the authors focus on the optimal number of customer orders which should be assigned to a batch such that the average turnover time is minimized. The optimal number depends on the storage policy and also on the pick times. A similar investigation of the average turnover time of a random customer order for a two block layout is carried out by Le-Duc & de Koster (2007). A corresponding model for fixed time window batching in a two block layout is presented by van Nieuwenhuyse & de Koster (2009).

All studies show that for variable time window batching the average turnover time of customer orders is a convex function of the number of orders per batch (batch size). A large batch size results in a small average processing time of each customer order. but also in a large average waiting time. On the other hand, the average (order) processing time is large for a small batch size, whereas the average waiting time is small. For fixed time window batching a similar convex function for the average turnover time of a customer order can be observed which is dependent on the length of the fixed time window (van Nieuwenhuyse & de Koster, 2009). Therefore, it can be concluded that an optimal batch size (or an optimal length of the fixed time window) exists which minimizes the average turnover time of a customer order. The function of the average turnover time in variable time window batching is depicted in Fig. 9. In numerical experiments, van Nieuwenhuysen & de Koster (2009) demonstrate that the application of fixed time window batching can lead to slightly smaller average turnover times than the application of variable time window batching.

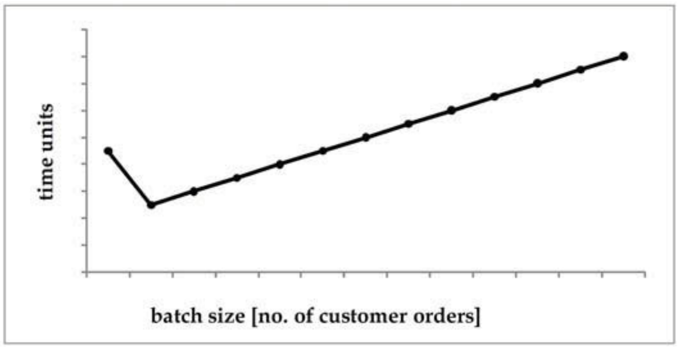

Fig. 9. Average turnover time against batch size for variable time window batching (based on Chew & Tang, 1999)

Extensions to the above-described travel time models may include multiple order pickers which can be modelled by additional queues (Le-Duc & de Koster, 2007). For time window batching, Van Nieuwenhuyse & de Koster (2009) present a combined analysis of a pick-and-sort system, in which the collected items are sorted and packed after the picking process. It is assumed that the time an order spends in the picking process is also stochastic. Turnover times now include the time in the picking process and the time in the sorting process. The authors concentrate on the allocation of workforce to both processes and aim at a minimization of the average turnover time.

Yu & de Koster (2009) describe an order picking area which is divided into several zones of identical size. Batches of customer orders are formed according to variable time window batching. The items of each batch are collected sequentially by zones, i.e. at first only items located in the first zone are picked, then all items located in the second zone, etc. The batch is complete when the items from the last zone have been picked. For this kind of systems, Yu & de Koster (2009) give an estimation of the average turnover times and observe that an optimal batch size exists.

9.3 Alternative Objective Functions and Approaches

Kamin (1998) describes a batching problem from practice, in which greeting cards have to be retrieved from a warehouse. Order pickers use automated guided vehicles on a fixed course collecting the items according to given customer orders. These customer orders arrive dynamically over time. Kamin focuses on the minimization of average turnover times. Therefore, the number of pick stops on the course is minimized. By means of a competitive analysis it is shown that every online algorithm is at least 2-competitive for this. Furthermore, the system is simulated and evaluated for several (simple) batching algorithms according to different evaluation criteria.

Henn (2010) describes an online order batching problem in a walk-and-pick warehouse in which the completion times of all (dynamically arriving) customer orders (or the makespan) are to be minimized. A competitive analysis reveals that any on-line algorithm for this problem is at least 2-competitive. The author also presents modifications of solution approaches for static order batching (FCFS, C&W(ii) and ILS) in order to deal with the dynamic situation.

10. Summary and Outlook on Future Research

In this paper we reviewed the literature dedicated to order batching in picker-to-parts order picking systems. We pointed out, that order batching is pivotal for the efficient management and control of order picking systems in distribution warehouses, since – due to the large amount of manual labour – order picking is the most cost-intensive function in a warehouse.
It has been shown that a large range of order batching methods exists and that research has evolved into several directions, of which static and dynamic batching are the predominant ones. Goals which may be used for the evaluation of solutions do not only include the minimization of processing times but also the earliness and tardiness of the customer orders, in particular when these orders have to be shipped at specific points in time.

We provided a comprehensive insight into order batching by giving a detailed state-of-the-art overview of the different solution approaches which have been suggested in the literature. Application of these methods can contribute to a significant improvement of the performance of the picking operations in warehouses.

As further research opportunities, research on static order batching could concentrate on the minimization of the total processing time for problems involving due dates. From a practical point of view it would also be desirable to intensify research on dynamic order batching and incorporate due dates in dynamic situations. Finally, we conclude that the interaction of order batching with the other related planning issues (layout design, item location, zoning, picker routing) has not been considered sufficiently in the literature so far. Thus, it might be worthwhile to provide corresponding simultaneous solution approaches in order to obtain a ″global″ optimum.

References

Albareda-Sambola, M.; Alonso-Ayuso, A.; Molina, E. & de Blas, C.S. (2009):
Variable Neighborhood Search for Order Batching in a Warehouse. *Asia-Pacific Journal of Operational Research 26*(5), 655-683.

Bozer, Y.A. & Kile, J.W. (2008):
Order Batching in Walk-and-Pick Order Picking Systems. *International Journal of Production Research* 46(7), 1887-1909.

Bullnheimer, B.; Hartl, R.F. & Strauss, C. (1999):
A New Rank Based Version of the Ant System - A Computational Study. *Central European Journal of Operations Research 7(1)*, 25-38.

Caron, F.; Marchet, G. & Perego, A., (2000):
Optimal Layout in Low-Level Picker-to-Part Systems. *International Journal of Production Research 38(1)*, 101–117.

Chen, M.-C. & Wu, H.-P. (2005):
An Association-Based Clustering Approach to Order Batching Considering Customer Demand Patterns. *Omega - International Journal of Management Science 33(4)*, 333-343.

Chew, E. & Tang, L. (1999):
Travel Time Analysis for General Item Location Assignment in a Rectangular Warehouse. *European Journal of Operational Research 112(3)*, 582-597.

Clarke, G. & Wright, J.W. (1964):
Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research 12(4)*, 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. *European Journal of Operational Research 182(2)*, 481-501.

de Koster, R.; Roodbergen, K.J. & van Voorden, R. (1999a):
Reduction of Walking Time in the Center of De Bijenkorf. In: *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(7)*, 1479-1504.

Drury, J. (1988):
Towards More Efficient Order Picking. The Institute of Materials Management: Cranfield.

Elsayed, E.A. (1981):
Algorithms for Optimal Material Handling in Automatic Warehousing Systems. *International Journal of Production Research 19(5)*, 525-535.

Elsayed, E.A. & Lee, M.-K. (1996):
Order Processing in Automated Storage/Retrieval Systems with Due Dates. *IIE Transactions 28(7),* 567-577.

Elsayed, E.A.; Lee, M.-K.; Kim, S. & Scherer, E. (1993):
Sequencing and Batching Procedures for Minimizing Earliness and Tardiness Penalty of Order Retrievals. *International Journal of Production Research 31(3)*, 727-738.

Elsayed, E.A. & Stern, R.G. (1983):
Computerized Algorithms for Order Processing in Automated Warehousing Systems. *International Journal of Production Research 21(4)*, 579-586.

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(7)*, 1097-1114.

Frazelle, E. (2002):
World-Class Warehousing and Material Handling, McGraw-Hill: New York.

Gademann, N. & van de Velde, S. (2005):
Order Batching to Minimize Total Travel Time in a Parallel-Aisle Warehouse. *IIE Transactions 37(1)*, 63-75.

Gademann, N.; van den Berg, J.P. & van der Hoff, H.H. (2001):
An Order Batching Algorithm for Wave Picking in a Parallel-Aisle Warehouse. *IIE Transactions 33(5)*, 385-398.

Garey, M.R. & Johnson, D.S. (1979):
Computers and Intractability: A Guide to the Theory of NP-completeness. W.H. Freeman and Co: New York.

Gibson, D.R. & Sharp, G.P. (1992):
Order Batching Procedures. *European Journal of Operational Research 58(1)*, 57-67.

Glover, F. (1986):
Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers and Operations Research 13(5)*, 533–549.

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

Hall, R.W. (1993):
Distance Approximations for Routing Manual Pickers in a Warehouse. *IIE Transactions 25(4)*, 76-87.

Henn, S. (2010):
Algorithms for On-line Order Batching in an Order-Picking Warehouse. *Proceedings of the 3rd International Conference on Information Systems, Logistics and Supply Chain ILS 2010,* Casablanca, April 2010, Business Process Consulting.

Henn, S.; Koch, S.; Doerner, K.; Strauss, C. & Wäscher, G. (2010):
Metaheuristics for the Order Batching Problem in Manual Order Picking Systems. *BuR - Business Research 3(1)*, 82-105.

Henn, S. & Wäscher, G. (2010):
Tabu Search Heuristics for the Order Batching Problem in Manual Order Picking Systems. *Working Paper No. 07/2010, Faculty of Economics and Management, Otto von Guericke University Magdeburg*, ISSN 1615-4274

Ho, Y.-C.; Su, T.-S. & Shi, Z.-B. (2008):
Order-Batching Methods for an Order-Picking Warehouse with two Cross Aisles. *Computers & Industrial Engineering 55(2)*, 321-347.

Ho, Y.-C. & Tseng, Y.-Y. (2006):
A Study on Order-Batching Methods of Order-Picking in a Distribution Center with two Cross Aisles. *International Journal of Production Research 44(17)*, 3391-3417.

Hsu, C.-M.; Chen, K.-Y. & Chen, M.-C. (2005):
Batching Orders in Warehouses by Minimizing Travel Distance with Genetic Algorithms. *Computers in Industry 56(2)*, 169-178.

Jarvis, J.M. & McDowell, E.D. (1991):
Optimal Product Layout in an Order Picking Warehouse. *IIE Transactions 23(1)*, 93-102.

Kamin, N. (1998):
On-Line Optimization of Order Picking in an Automated Warehouse. Shaker: Aachen.

Le-Duc, T. & de Koster, R. (2007):
Travel Time Estimation and Order Batching in a 2-Block Warehouse. *European Journal of Operational Research 176(1)*, 374-388.

Pan, J.C.-H. & Liu, S. (1995):
A Comparative Study of Order Batching Algorithms. *Omega - International Journal of Management Science 23(6)*, 691-700.

Petersen, C.G. & Schmenner, R.W. (1999):
An Evaluation of Routing and Volume-based Storage Policies in an Order Picking Operation. *Decision Sciences 30(2)*, 481-501.

Pohl, L.; Meller, R. & Gue, K. (2009):
Optimizing Fishbone Aisles for Dual-Command Operations in a Warehouse. *Naval Research Logistics 56(5)*, 389-403.

Ratliff, H.D. & Rosenthal, A.R. (1983):
Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem. *Operations Research 31(3)*, 507-521.

Roodbergen, K.J. & de Koster, R. (2001):
Routing Methods for Warehouses with Multiple Cross Aisles. *International Journal of Production Research 39(9)*, 1865-1883.

Rosenwein, M.B. (1996):
A Comparison of Heuristics for the Problem of Batching Orders for Warehouse Selection. *International Journal of Production Research 34(3)*, 657-664.

Ruben, R.A. & Jacobs, F.R. (1999):
Batch Construction Heuristics and Storage Assignment Strategies for Walk/Ride and Pick Systems. *Management Science 45(4)*, 575-596.

Tompkins, J.A.; White, J.A.; Bozer, Y.A.; Frazelle, E. & Tanchoco, J.M.A. (2003):
Facilities Planning. 3rd ed., Wiley: 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(22)*, 6533-6555.

van Nieuwenhuyse, I. & de Koster, R. (2009):
Evaluating Order Throughput Time in 2-block Warehouses with Time Window Batching. *International Journal of Production Economics 121*, 654-664.

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

Whittley, I. & G. Smith (2004):
The Attribute Based Hill Climber. *Journal of Mathematical Modelling and Algorithms 3(2)*, 167–178.

Won, J. & Olafsson, S. (2005):
Joint Order Batching and Order Picking in Warehouse Operations. *International Journal of Production Research 43(7)*, 1427-1442.

Yu, M. & de Koster, R. (2009):
The Impact of Order Batching and Picking Area Zoning on Order Picking System Performance. *European Journal of Operational Research 198(2)*, 480-490.

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.ww.uni-magdeburg.de

ISSN 1615-4274