OTTO-VON-GUERICKE-UNIVERSITY MAGDEBURG
FACULTY OF ECONOMICS AND MANAGEMENT

Algorithms for On-line Order Batching in an Order-Picking Warehouse

Sebastian Henn

FEMM Working Paper No. 34, October 2009

F E M M

Faculty of Economics and Management Magdeburg

Working Paper Series

Otto-von-Guericke-University Magdeburg
Faculty of Economics and Management
P.O. Box 4120
39016 Magdeburg, Germany
http://www.ww.uni-magdeburg.de/

Impressum (§ 5 TGM):

Herausgeber:
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Die Dekanin

Verantwortlich für diese Ausgabe: Gerhard Wäscher
Otto-von-Guericke-Universität Magdeburg
Postfach 4120
39016 Magdeburg

http://www.ww.uni-magdeburg.de/fwwdeka/femm/

Auflage: (50)

Redaktionsschluss: Oktober 2009

Herstellung: Dezernat Allgemeine Angelegenheiten,
Sachgebiet Reproduktion

Bezug über den Herausgeber

ISSN 1615-4274

Algorithms for On-line Order Batching in an Order-Picking Warehouse

Sebastian Henn*

Abstract

In manual order picking systems, order pickers walk or ride through a distribution warehouse in order to collect items required by (internal or external) customers. Order batching consists of combining these – indivisible – customer orders into picking orders. With respect to order batching, two problem types can be distinguished: In off-line (static) batching all customer orders are known in advance. In on-line (dynamic) batching customer orders become available dynamically over time. This report considers an on-line order batching problem in which the total completion time of all customer orders arriving within a certain time period has to be minimized. The author shows how heuristic approaches for the off-line order batching can be modified in order to deal with the on-line situation. A competitive analysis shows that every on-line algorithm for this problem is at least 2-competitive. Moreover, this bound is tight if an optimal batching algorithm is used. The proposed algorithms are evaluated in a series of extensive numerical experiments. It is demonstrated that the choice of an appropriate batching method can lead to a substantial reduction of the completion time of a set of customer orders.

Keywords: Warehouse Management, Order Picking, Order Batching, On-line Optimization

*Otto-von-Guericke University Magdeburg, Faculty of Economics and Management

Corresponding author:
Dipl.-Math. oec. Sebastian Henn
Otto-von-Guericke-University Magdeburg
Faculty of Economics and Management
Department of Management Science
Postbox 4120
39106 Magdeburg
sebastian.henn@ovgu.de

Contents

1 Introduction

Order Picking is a warehouse function dealing with the retrieval of items (articles) from their storage locations in order to satisfy (internal or external) customer orders. It arises because incoming articles are received and stored in (large volume) unit loads while customers tend to order small volumes of different articles (de Koster et al., 2007). Order picking is critical to each supply chain, since underperformance results in unsatisfactory customer service (long processing and delivery times) and high costs (labor costs, costs of additional and/or emergency shipments). Even though different attempts have been made to automate the picking process, systems involving human operators are still prevalent in practice. Such manual order picking systems can be differentiated into two categories (Wäscher, 2004): Picker-to-parts systems, where order pickers drive or walk through the warehouse and collect the required items; and parts-to-picker systems, where automated storage and retrieval systems deliver the items to stationary order pickers. In systems of the first kind, which are considered in this paper, three activities at the operative level can be distinguished (Caron et al., 1998): the assignment of items to storage locations (item location), the transformation of customer orders into picking orders (order batching) and the routing of pickers through the warehouse (picker routing). This paper stresses the second activity where different customer orders can be combined into picking orders (batches) and jointly released for picking. This activity has proven to be pivotal for the efficiency of warehouse operations (de Koster et al., 1999).

With respect to the availability of the customer orders, two situations for batching customer orders can occur (Yu and de Koster, 2009): In off-line (static) batching all customer orders are known at the beginning of the (short term) planning period (shift or day). In on-line (dynamic) batching customer orders become available dynamically over time. Batches have to be formed based only on the known customer orders. The aim of this paper is to examine how solution approaches for the static batching problem can be modified for on-line situations in order to improve the warehouse efficiency. Moreover, decision rules are proposed that define, which customer orders should be satisfied directly, and which ones should be satisfied later, if on a specific point in time more than one batch can be released.

The remainder of this paper is organized as follows: In Section 2 the On-line Order Batching Problem will be defined and an optimization model will be given. Section 3 contains an overview of the relevant literature. Algorithms for the On-line Order Batching Problem will be presented in Section 4. To analyze the solution quality of these algorithms and to show the limitations for possible algorithms, a competitive analysis of the generated solutions, will be performed in Section 5. Moreover, extensive numerical experiments have been carried out to evaluate the performance of the proposed algorithms. Purpose and design of the numerical study will be described in Section 6. The test results will be compared for different problem classes in Section 7. The paper will conclude with a summary and an outlook on further research topics.

2 On-line Order Batching Problem

2.1 Problem Description

In a manual picker-to-parts system *order pickers* are guided by *pick lists*, which specifies the sequence in which the storage locations should be visited as well as the number of items demanded of each article. A pick list may contain the items of a single customer order (pick-by-order) or of a combination of customer orders (pick-by-batch). The picking process can be described in the following way: The order picker starts at the *depot*, walks (or rides on an appropriate vehicle) through the warehouse and collects items from different storage locations. Afterwards, he/she returns to the depot and hands over the picked items. The corresponding route through the warehouse is typically determined by means of a so-called routing strategy. Despite the fact that an optimal, polynomial time algorithm for the picker routing problem exists (Ratliff and Rosenthal, 1983), it is hardly ever used in practice. Order pickers do not seem to accept the optimal routes, since they are not always straightforward and sometimes even confusing (de Koster et al., 1999). Two well known strategies are the *S-Shape* and the *Largest Gap* heuristic which provide the required non confusing routing schemes. Figure 2.1 demonstrates the straightforward character of both routing schemes for a set of items to be picked. The black rectangles symbolize the corresponding locations where items have to be picked (pick locations).
The *S-Shape Heuristic* provides solutions in which the order picker enters and traverses an aisle completely if at least one required item is located in that aisle (an exception would be the last aisle if the order picker is positioned on the front cross-aisle, i.e. the cross-aisle with the depot). Afterwards, the order picker moves to the next aisle which has to be visited (Hall, 1993). The *Largest Gap Heuristic* gives a solution in which the order picker completely traverses the first aisle and the last aisle containing a demanded item. All other aisles – containing at least one required item – are entered from the front and from the back in a way that the non-traversed distance between two adjacent pick locations or the end of the aisle is maximal.
Order picking is usually done with the help of a picking device (e.g. cart, roll pallet etc.). Conse-

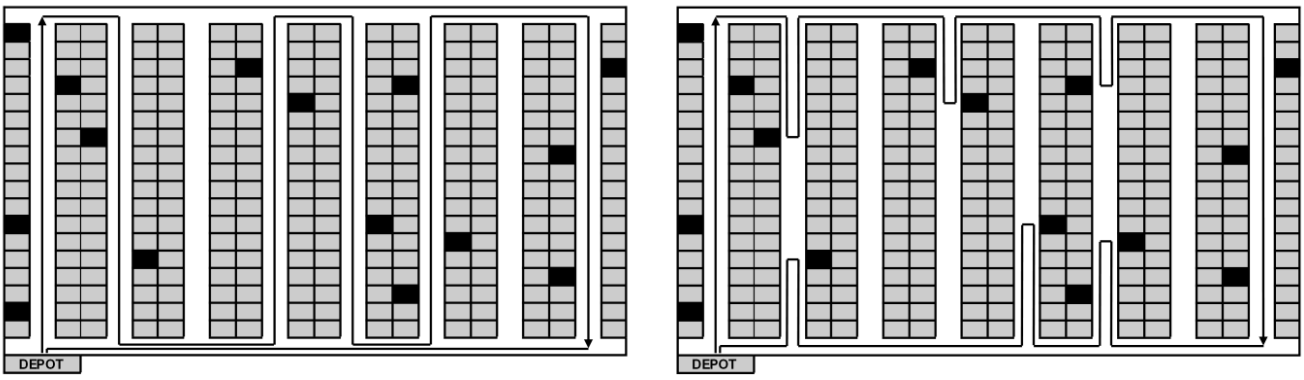


Figure 2.1: Example of S-Shape (left) and Largest Gap (right) heuristic in a single-block warehouse

quently, customer orders can be combined until the capacity of the picking device is exhausted. This capacity is typically defined by a number of items. The splitting of a customer order into two or more batches is prohibited, since it would result in additional unacceptable sorting effort. If an order picker has started a tour through the warehouse, an interruption of this process and a rearrangement of the customer orders is also not permitted.
The time period necessary to complete a batch is called (batch) *service time* (batch *processing time*). It is composed of the *travel time*, i.e. the time period the order picker needs to travel from the depot to the first pick location, between the pick locations and from the last pick location to the depot; the *search time*, i.e. the time period needed for the identification of articles; the *pick time*, i.e. the time period needed for moving the items from the pick location on the picking device; and the *setup time*, i.e. the time period for administrative and set-up tasks at the beginning and end of each tour (Chew and Tang, 1999).
In the situation considered here the customer orders are not known in advance, but become available over time. At a specific point in time, it is only known if at least one customer orders will arrive in future. However, no information about how many orders or their characteristics will arrive is given. The decision, which customer orders should be processed directly, has to be made without considering information of future incoming orders.
The point in time, when a customer order becomes available, is called *arrival time*. The *start time* (*release time*) of a batch is the point in time when an order picker starts to process this batch. The start time of an order is identical to the start time of the batch the order is assigned to. The point in time when the order picker returns to the depot after collecting all items is called *completion time* of a batch or of an customer order, respectively.
The (customer order) *waiting time* can be determined as the length of the time period between the arrival time and the start time of a customer order. The *turnover time* (*response time*) is the time period for which a customer order stays in the system, i.e. the time period between the completion and the start time of a customer order.
If the number of arriving customer orders is too large for processing each customer order separately in an appropriate total time, customer orders must be combined to batches. The *On-line Order Batching Problem* consists of grouping customer orders into batches such that the completion time of all orders – identical to the completion time of the last released batch – is minimized. In the following, we discuss the situation with a single order picker, i.e. all batches must be processed one after another.

2.2 Optimization Model

We now formulate an optimization model for the off-line version of the *Order Batching Problem* described above. The model, which requires the complete information of all incoming orders, is presented in order to analyze the structure of the problem. The following constants are used:

n: number of customer orders;
m: number of batches;

r_i: arrival time of customer order i for all $i \in \{1, \ldots, n\}$, where $0 \leq r_i \leq r_{i+1}$ holds;
t_{setup}: setup time, i.e. time necessary for administrative and set-up tasks for each batch;
v_{travel}: travel velocity, i.e. number of length units the order picker can cover in the warehouse per time unit;
v_{pick}: pick velocity, i.e. number of items the order picker can search and pick per time unit;
w_i: number of items of customer order i ($\forall i \in \{1, \ldots, n\}$);
W: maximal umber of items can be included in a batch.

The model uses the following variables.

s_j: start time of batch j ($\forall j \in \{1, \ldots, m\}$);
$\mathbf{x}_j$: $= (x_{j1}, \ldots, x_{jn})^T$ a vector in $\mathbb{B}^n$ where

$$x_{ji} = \begin{cases} 1 & \text{if customer order } i \text{ is assigned to batch } j \\ 0 & \text{else} \end{cases} \quad \text{for all } j \in \{1, \ldots, m\}.$$

Without loss of generality, we assume that batch j is started after batch $j-1$ has been completed, i.e. $s_j > s_{j-1}$. The total length of the picking tour for a particular batch involving a particular routing method is determined by the function $d : \mathbb{B}^n \mapsto \mathbb{Q}$. For an empty batch the function value is zero, i.e. $d(\mathbf{0}) = 0$ where $\mathbf{0} = (0, \ldots, 0)^T$. The problem can be modelled as follows:

$$\min \max_{j \in \{1,\ldots,m\}} \{s_j + \frac{d(\mathbf{x}_j)}{v_{\text{travel}}} + \frac{\sum_{i=1}^{n} w_i x_{ji}}{v_{\text{pick}}} + t_{\text{setup}} | \text{at least one } x_{ji} = 1\} \tag{2.1}$$

$$\text{s.t.} \sum_{j=1}^{m} x_{ji} = 1, \quad \forall i \in \{1, \ldots, n\} \tag{2.2}$$

$$\sum_{i=1}^{n} w_i x_{ji} \leq W, \quad \forall j \in \{1, \ldots, m\} \tag{2.3}$$

$$s_j \geq \max_{i \in \{1,\ldots,n\}} \{r_i \cdot x_{ji}\}, \quad \forall j \in \{1, \ldots, m\} \tag{2.4}$$

$$s_j \geq s_{j-1} + \frac{d(\mathbf{x}_{j-1})}{v_{\text{travel}}} + \frac{\sum_{i=1}^{n} w_i x_{j-1,i}}{v_{\text{pick}}} + t_{\text{setup}}, \quad \forall j \in \{2, \ldots, m\} \tag{2.5}$$

$$s_j \geq 0, \quad \forall j \in \{1, \ldots, m\} \tag{2.6}$$

$$\mathbf{x}_j \in \mathbb{B}^n, \quad \forall j \in \{1, \ldots, m\} \tag{2.7}$$

In the objective function (2.1) the expression $d(\mathbf{x}_j)/v_{\text{travel}} + \sum_{i=1}^{n} w_i x_{ji}/v_{\text{travel}} + t_{\text{setup}}$ represents the service time of a batch. This sum is composed of the time the order picker needs to travel through the warehouse, the time he/she needs to pick the items and the setup time. By addition of the start time s_j of the batch j, its completion time is obtained. In summary, (2.1) minimizes the maximal completion time of all batches with at least one assigned customer order. Equations (2.2) ensure the assignment of each customer order to exactly one batch. Furthermore, inequalities (2.3) guarantee that the capacity of the picking device is not violated. The conditions (2.4) indicate that a batch is started after all customer orders assigned to this batch are known. From (2.5) follows that a batch is started after the previous one is completed. Finally, the constraints of

types (2.6) and (2.7) indicate that start times are non-negative and the variables $\mathbf{x}_j$ are binary vectors, respectively.
To verify the model, it remains to show that empty batches do not affect the objective function value. This is necessary since (2.5) causes that $s_j > s_{j-1}$ if t_{setup} is set greater than 0. This inequality also holds in the case when no customer order is assigned to batch $j-1$. Suppose an instance for this problem, in which only $\tilde{m}$ batches ($\tilde{m} < m$) are needed to process all customer orders. Without loss of generality, we assume that $s_1 < s_2 < \cdots < s_{\tilde{m}-1} < s_{\tilde{m}}$ holds for the start time of these batches. Then we set the start times $s_{\tilde{m}+k}$ to $s_{\tilde{m}+k-1} + t_{\text{setup}}$ and $x_{i,\tilde{m}+k}$ to 0 for all $k \in \{1, \ldots, m-\tilde{m}\}$ and for all $i \in \{1, \ldots, n\}$. This solution satisfies all constraints. Their objective function value is the completion time of batch $\tilde{m}$ which is equal to the optimal objective function value.

3 Literature Review

With respect to order batching, research can be differentiated into two types according to the availability of data: *off-line (static) batching* and *on-line (dynamic) batching* (Yu and de Koster, 2009). In off-line batching complete information of all orders is given at the beginning of the planning period. The on-line type considers the stochastic arrival process.
For off-line order batching Gademann and van de Velde (2005) and Gademann et al. (2001) show the $\mathcal{NP}$-hardness of the problem of minimizing the total travel time and respectively minimizing the maximal service time for any of the batches, if the number of customer orders per batch is greater than two.
For the off-line Order Batching Problem, where the objective is to minimize the total traveling time, Gademann and van de Velde (2005) formulate an optimization model. They present also a branch-and-price algorithm with column generation that was able to solve small instances to optimality in reasonable computing time. Their formulation is independent from the assumed routing strategy like the formulation given here in Section 2. For the case of S-Shape routing, Bozer and Kile (2008) present a mixed integer programming approach, that generates near optimal solutions for small sets of customer orders (up to 25).
Chen and 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. The following 0-1 integer programming approach is applied to cluster the customer orders into batches.
For larger problems the use of heuristics is still advisable. These heuristic approaches can be distinguished in four groups. The first ones are *priority rule-based algorithms*, where customer orders are ranked according to a priority value and then are assigned to batches following this rank (Gibson and Sharp, 1992). The probably best-known and straightforward way is the application of the First-Come-First-Served rule (FCFS). Other priority rules include space-filling curves (Gibson and Sharp, 1992; Pan and Liu, 1995). The assignment of customer orders to batches can either be done sequentially (Next-Fit-Rule) or simultaneously (First-Fit-Rule, Best-Fit-Rule)

(Wäscher, 2004). The second group consists of *seed algorithms*, introduced by Elsayed (1981) and Elsayed and Stern (1983), which generate batches sequentially. They select one customer order as a start order for a batch. Additional customer orders are assigned to that batch according to an order-congruency rule. An overview of the various seed selection and order-congruency rules is given by Ho et al. (2008). Methods of the third group, *savings algorithms*, are based on the Clarke-and-Wright-Algorithm for the Vehicle Routing Problem (Clarke and Wright, 1964) and have been adapted in several ways for the Order Batching Problem. For each pair of customer orders, the savings can be obtained by collecting the items of the two customer orders in one (large) tour instead of collecting them in two separate tours (de Koster et al., 1999; Elsayed and Unal, 1989). Starting with the pair of customer orders with the largest savings, the pairs are considered for being assigned to a batch in a non-ascending order. Finally, the last group contains *metaheuristics*. Hsu et al. (2005) present a Genetic Algorithm for the Order Batching Problem. Their approach includes an aisle-metric for the determination of the tour lengths and is, therefore, limited to S-Shape-Routing, only. Tsai et al. (2008) describe an integrated approach, in which solutions to the batching problem as well as to the routing problem are determined by a Genetic Algorithm. Iterated Local Search and a variant of Ant Colony Optimization are applied to the Order Batching Problem by Henn et al. (2009).

With respect to on-line batching Kamin (1998) describes a real world problem, in which greeting cards have to be retrieved from a warehouse. Order pickers use automated guided vehicles on a fixed course to collect greeting cards according to customer orders. Besides a theoretical analysis, the system is simulated and evaluated according to different objectives including the time needed to complete all customer orders of a day. The impact of different batching strategies and problem parameters is investigated. In contrast to our problem, Kamin selects the next batch which should be processed according to a due date, i.e. a point in time when the customer order is due to be completed.

Apart from this approach, *time window batching* is prevalent in the on-line situation (Van Nieuwenhuyse and de Koster, 2009). Time window batching can be carried out in two different variants, namely fixed and variable time window batching. In fixed time window batching all orders arriving during a particular time interval are assigned to one batch. In variable time window batching the order picker waits until a particular number of orders has arrived, and collect the items of these orders in a joint tour.

Chew and Tang (1999); Tang and Chew (1997) describe an on-line problem in which the number of order pickers is limited. They carry out a theoretical analysis of travel and service times on the basis of S-Shape routing. To measure the quality of their estimation the authors simulate the picking system 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. They use variable time window batching where each batch consists of exactly n_0 customer orders. If n_0 customer orders are 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. For a 2-block warehouse, Le-Duc (2005) and Le-Duc and de Koster (2007) prove an estimation for the average turnover time for a random customer order and observe similar results like Chew and Tang with an uniformly distributed demand frequency. The authors conclude that the average turnover time of customer orders is a convex function of the number of orders per batch (batch size). On the one hand, a large batch size leads to a small average service time of each customer order, but to a large average waiting time. On the other hand, the average service time is large for a small batch size, whereas the average waiting time is small. The authors conclude that an optimal batch size exists. Le-Duc and de Koster (2007) also mention possible extensions to this model, i.e. multiple order pickers, which can be modeled via different queues and multi-line orders. A queuing model to determine the average turnover time for a customer order in a 2-block warehouse for variable and fixed time window batching is given by Van Nieuwenhuyse and de Koster (2009).

A joint batching and picking problem where customer orders arrive at different times is given by Won and Olafsson (2005). Like in the previously mentioned articles, the authors observe a trade-off between the total service time and the turnover time of a customer order. They formulate an optimization model where the objective function is a weighted sum of service and waiting time of customer orders within a batch. For this approach the arrival times of the customer orders must be known in advance. To solve the problem the authors propose a two step procedure in which at first customer orders are batched by means of the FCFS rule. The subsequent routing problem is then solved by a 2-opt procedure. In the numerical experiments the authors assume a warehouse with low order volumes and search for the ideal waiting time of a batch. They also present an improvement heuristic to reduce the turnover times by combining customer orders.

Elsayed and Lee (1996) describe an automated storage and retrieval system where some articles have to be picked from the warehouse (retrieval orders) and some have to be stored in the warehouse (storage orders). The arrivals of the retrieval orders are dynamic and due dates have been assigned to each order. The objective function is meant to form batches and sequence them in a way that the tardiness (maximum of the completion time minus the due date and zero) of the customer orders is minimized. The authors distinguish between a static and a dynamic case. In the static case, customer orders arriving in a particular time interval form a group. This time interval is determined by the time necessary to process the customer orders of the previous group. In the dynamic case, a customer order arriving while a group of customer orders is being processed is added to the set of non-processed customer orders. This results in a new set of batches. In their approach, customer orders are sequenced according to their due dates and the times needed to process the customer orders in a single batch. According to this sequence, three decision rules are proposed and evaluated for the selection of batches: a nearest schedule rule, a shortest service time rule, and a most common locations rule.

4 Algorithms

4.1 Basic Principle

The difference between on-line and off-line problems consists in the availability of the input parameters. Formally, an instance of an on-line problem can be described as a (input) sequence of requests. This is in the On-line Order Batching Problem a sequence of customer orders, $1, \ldots, n$, with different arrival times $r_1, \ldots, r_n$. An *on-line algorithm* applied to this sequence has to deal with each request i at time r_i, independent from the requests $i+1, \ldots, n$. An algorithm for the On-line Order Batching Problem has to form and release batches without having complete information on the types and the arrival times of future customer orders.
The points in time when a decision of this kind has to be made are called *decision points*. These can be distinguished into three classes. A decision of the first kind is one point when a set of unprocessed (also called *open*) customer orders exists and an order picker becomes available. Decision points of this type appear at the beginning of the planning period or at the completion time of a batch. At this point either the next batch should be released directly or its start should be postponed to a later point in time. A decision point of the second type is a point in time when an order picker is idle and a new customer order arrives. The algorithm can rearrange the set of batches and determine a start time for the next batch. Decision points of the third type are the

Algorithm 4.1 Basic principle of an on-line order picking algorithm

set $t = 0$, $P(0) = \{i \in \{1, \ldots, n\} | r_i = 0\}$;
repeat
  generate a set of batches $B(t)$ by means of batching heuristic $\mathcal{H}_b$ to $P(t)$;
  if $|B(t)| = 0$ **then**
    set t to the arrival time of the next customer order;
  else if $|B(t)| = 1$ **then**
    let j' be the batch in $B(t)$;
    $i' = \arg\max\{st_i | i \text{ is assigned to } j'\}$;
    if $t < 2r_{i'} + st_{i'} - st_{j'}$ and the last customer order is not known **then**
      set t to the minimum of $2r_{i'} + st_{i'} - st_{j'}$ and the arrival time of the next order;
    else
      start batch j' and set t to $t + st_j$;
    end if
  else
    select one batch j of $B(t)$ according to selection rule $\mathcal{H}_s$;
    start batch j;
    set t to $t + st_j$;
  end if
  update $P(t)$;
until no further customer order arrives and $P(t)$ is empty;

points in time when the last customer order arrives. At this point the order picker can start all remaining batches one after another.
The basic principle of the proposed on-line algorithm combines ideas of Kamin (1998) for a special warehouse type and of Zhang et al. (2001) for the related on-line scheduling problem of minimizing the makespan on a batch processing machine. (According to the three field classification for scheduling problems their problem is classified as $1/r_j, B, \text{on-line}/C_{\max}$.) At each decision point t we determine a solution for the off-line version of the Order Batching Problem for all customer orders which are known but have not been satisfied at that time. The set $P(t)$ of these customer orders is called *set of open orders*. In contrast to the scheduling problem mentioned by Zhang et al. (2001) the off-line version of the Order Batching Problem is $\mathcal{NP}$-hard (Gademann et al., 2001). Therefore, the application of a batching heuristic $\mathcal{H}_b$ is suggested in order to obtain a set of batches $B(t)$. From this set a batch is selected and released according to a selection rule $\mathcal{H}_s$. An exception would be the case where a solution for batching $P(t)$ leads only to a single batch j' with service time $st_{j'}$. Let i' be a customer order assigned to j' which would result in the longest service time $st_{i'}$, if each customer order was processed separately. If $2r_{i'} + st_{i'} - st_{j'}$ is greater or equal than the actual time t the batch j' is released directly. Otherwise, this batch is started at time $2r_{i'} + st_{i'} - st_{j'}$, unless a new customer order arrives in the meantime. This case would be a decision point of the second kind and a new set of batches $B(t)$ would be determined. In case of the arrival of the last customer order, all batches are started sequentially. Algorithm 4.1 summarizes this approach.

4.2 Batching Heuristics

For the batching heuristic $\mathcal{H}_b$ we consider three options, namely the *First-Come-First-Served* (FCFS) rule, the *savings algorithm* C&W(ii) and *Iterated Local Search* (ILS).
In FCFS the customer orders are ranked according to their arrival time. With respect to the capacity constraint and this list the customer orders are assigned to batches.
In the *savings algorithm* C&W(ii) at the beginning each customer order is assigned to a different batch (de Koster et al., 1999; Elsayed and Unal, 1989). The savings are computed for each combination of batches. These values can be obtained by collecting the items of both batches in one (large) tour instead of collecting them in two separate tours. The pairs of batches are ranked according to their savings. The pair with largest savings will be combined if it does not violate the capacity constraint. Otherwise, the pair with the second largest savings is considered. This step is repeated until a combination is possible or no pair exists with positive savings which can be combined. In the latter case the algorithm stops. In case that two batches were combined, the savings are computed again, and one searches for a further combination of batches.
Iterated Local Search has been successfully applied to the Order Batching Problem (Henn et al., 2009). It is based on the Local Search principle, where one starts from a candidate solution and searches iteratively for better neighbor solutions. ILS consists of two alternating phases, an improvement and a perturbation phase. In the first phase one starts from an initial solution and terminates in a local optimum. The vicinity of this local optimum (used as an incumbent

solution) will be explored in order to identify a solution with an improved objective function value. Therefore, in the perturbation phase, the incumbent solution is partially destroyed and a further improvement search phase is applied to this solution. This new local optimum has to pass an acceptance criterion in order to become the new incumbent solution, otherwise the previous solution remains the incumbent solution for a further perturbation. These two phases are repeated until a termination condition is met. For the detailed application of ILS we refer to Henn et al. (2009).

4.3 Selection Rules

For the selection rule $\mathcal{H}_s$ four different options are suggested. The Selection Rule 4.1 FIRST chooses the first batch of $B(t)$. If batching is carried out by means of the FCFS rule, this rule selects a batch whose customer orders have minimal arrival times among all open customer orders.

Selection Rule 4.1 FIRST
select the first batch of $B(t)$;

The following selection rules are meant to control the choice of a specific batch in a way that will influence the set of customer orders and subsequent decisions. Selection Rule 4.2 SHORT determines a batch with smallest service time. Batches with a small service time may not include customer orders which guide the order picker to aisles far from the depot. If the demand frequency of articles is low in those aisles it may be preferable to collect customer orders, demanding these items, and process them together at the end of the planning period.

Selection Rule 4.2 SHORT
select the batch of $B(t)$ with smallest service time;

As opposed to the previous strategy Selection Rule 4.3 LONG selects the batch which requires the longest service time. With this rule the time interval between two batching steps is very large and while the batch is processed new customer orders may arrive which can be combined more favorably with the non-processed customer orders.

Selection Rule 4.3 LONG
select the batch of $B(t)$ with longest service time;

The Selection Rule 4.4 SAV computes for each batch a savings value, i.e. the sum of the single service times of the assigned customer orders and subtracts the total service time of the batch. By selecting the batch with the largest sum, one intends to release a batch with similar orders.

Selection Rule 4.4 SAV

for all batches in $B(t)$ **do**

determine the savings value, i.e. the sum of the service times if each customer order of the batch is processed in a single batch and subtract this value from the service time of the batch;

end for

select the batch with largest savings value, in the case of more than one batch with largest savings value, select the batch with lowest index;

5 Competitive Analysis

5.1 Scope

In the context of on-line optimization, the theoretical investigation of the computational complexity of algorithms may be unappropriate for algorithms dealing with uncertainty (Fiat and Woeginger, 1998). Also, a traditional worst-case analysis concerning the solution quality of an on-line algorithm may not be appropriate. Instead, *competitive analysis* is a common approach for the evaluation of the performance of an on-line algorithm. It is based on the considerations of Sleator and Tarjan (1985) and analyzes the performance on each input sequence. This is done by comparison of an on-line algorithm with an optimal off-line algorithm, i.e. an algorithm that determines an optimal solution for the complete input sequence. More formally, an on-line algorithm $\mathcal{A}$ for a minimization problem is called c-competitive if a constant α exists such that for all possible input sequences $\mathcal{I}$ the inequality $\mathcal{A}(\mathcal{I}) \leq c \cdot \text{OPT}(\mathcal{I}) + \alpha$ holds, where $\mathcal{A}(\mathcal{I})$ is the objective function value provided by algorithm $\mathcal{A}$ for instance $\mathcal{I}$ and $\text{OPT}(\mathcal{I})$ is the objective function value of an optimal off-line solution for $\mathcal{I}$. The infimum over all c for which the inequality holds is called *competitive ratio*.

For the On-line Order Batching Problem we show that the algorithms presented in Section 4.1 are at least 2-competitive for each combination of a batching heuristic with a selection rule. Additionally, we show that this bound is tight for an optimal batching approach, independently of the used selection rule. We assume a single-block warehouse with two cross aisles, one in the front and one in the back of the picking area. The depot is located in front of the leftmost (picking) aisle and all (picking) aisles are vertically orientated. The layout of the warehouse is depicted in Figure 5.1. The warehouse consists of U aisles with C storage locations (cells) on each side of an aisle. Being positioned in the center of an aisle, the order picker can pick items from cells on the right, as well as from the cells on the left without additional movements. Whenever the order picker leaves an aisle he/she has to move in order to reach the cross aisle a distance equal to the width of a cell l_{cw} in vertical direction from the first storage location, or from the last storage location, respectively. Let l_c denote the center-to-center distance between two aisles, i.e. the distance an order picker has to move in horizontal direction from one aisle to the next aisle.

5.2 Lower Bound

Selection Rules FIRST, LONG, SAV

In order to obtain the proposed lower bound for the competitive ratio, we first describe an example for the case that Algorithm 4.1 is applied with an arbitrary batching heuristic and the selection rules FIRST, LONG or SAV. We assume that the number of storage locations on a side of an aisle C is greater than $\max\{2, (U-1)\frac{l_c}{l_{cw}} + 1\}$ and that the maximal number W of items in a batch, is even. Let two different customer orders i_1 and i_2 be available at time 0, of which i_1 requires $\frac{W}{2} + 1$ items stored in the farmost cell on the left of the leftmost aisle. The second customer order i_2 demands $\frac{W}{2} + 1$ items of the article stored in the first cell of the rightmost aisle. Since both customer orders cannot be included in the same batch, two batches are needed for completing i_1 and i_2. The routing of the order picker for both batches according to the S-Shape heuristic and the Largest Gap strategy, respectively, is shown in Figure 5.1. The batch containing order i_1 has

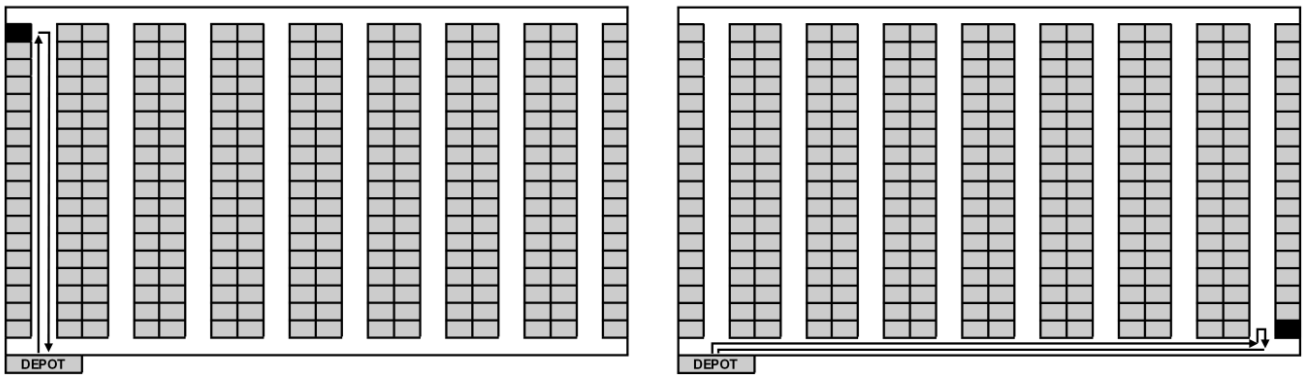


Figure 5.1: Routing for the customer orders i_1 (left) and i_2 (right)

the service time of

$$st_{i_1} = \frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{\frac{W}{2} + 1}{v_{\text{pick}}} + t_{\text{setup}},$$

the second one containing i_2 has the service time

$$st_{i_2} = \frac{2 \cdot (U-1) \cdot l_c + 2 \cdot 1 \cdot l_{cw}}{v_{\text{travel}}} + \frac{\frac{W}{2} + 1}{v_{\text{pick}}} + t_{\text{setup}}.$$

According to Algorithm 4.1, the order picker processes one of these batches immediately. By the application of selection rules FIRST, LONG, and SAV the batch with customer order i_1 will be processed first. We now consider the problem instance where at time 1 a customer order i_3 becomes available, i.e. when the order picker has just started to collect the items of customer order i_1. This customer order i_3 is similar to the first customer order i_1 and requires $W - (\frac{W}{2} + 1) = \frac{W}{2} - 1$ items, of which all are located in the leftmost aisle, while one of them is stored in the last cell of the aisle. The on-line algorithm generates a solution in which customer orders i_2 and i_3 are

assigned to one batch with a completion time of

$$\begin{aligned}\mathcal{A}_{4.1}(\mathcal{I}) &= st_{i_1} + st_{\{i_2,i_3\}} \\ &= \frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{\frac{W}{2}+1}{v_{\text{pick}}} + t_{\text{setup}} \\ &\quad + \frac{2 \cdot (U-1) \cdot l_c + 2 \cdot (C+1) \cdot l_{cw}}{v_{\text{travel}}} + \frac{\frac{W}{2}+1+W-(\frac{W}{2}+1)}{v_{\text{pick}}} + t_{\text{setup}} \\ &= \frac{2(U-1) \cdot l_c + (4 \cdot C+2) l_{cw}}{v_{\text{travel}}} + \frac{W+\frac{W}{2}+1}{v_{\text{pick}}} + 2 \cdot t_{\text{setup}}.\end{aligned}$$

An optimal off-line algorithm generates a solution in which customer order i_2 is processed first and, afterwards, the customer orders i_1 and i_3 are processed in one batch together. This leads to

$$\begin{aligned}\text{OPT}(\mathcal{I}) &= st_{i_2} + st_{\{i_1,i_3\}} \\ &= \frac{2 \cdot (U-1) \cdot l_c + 2 \cdot 1 \cdot l_{cw}}{v_{\text{travel}}} + \frac{\frac{W}{2}+1}{v_{\text{pick}}} + t_{\text{setup}} \\ &\quad + \frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{\frac{W}{2}+W-(\frac{W}{2}+2)}{v_{\text{pick}}} + t_{\text{setup}} \\ &= \frac{2(U-1) \cdot l_c + 2(C+1) \cdot l_{cw}}{v_{\text{travel}}} + \frac{W+\frac{W}{2}+1}{v_{\text{pick}}} + 2 \cdot t_{\text{setup}}.\end{aligned}$$

In order to provide a lower bound for the competitive ratio we consider

$$\begin{aligned}\frac{\mathcal{A}_{4.1}(\mathcal{I})}{\text{OPT}(\mathcal{I})} &= \frac{\frac{2(U-1) \cdot l_c + (4 \cdot C+2) \cdot l_{cw}}{v_{\text{travel}}} + \frac{W+\frac{W}{2}+1}{v_{\text{pick}}} + 2 \cdot t_{\text{setup}}}{\frac{2(U-1) \cdot l_c + 2(C+1) \cdot l_{cw}}{v_{\text{travel}}} + \frac{W+\frac{W}{2}+1}{v_{\text{pick}}} + 2 \cdot t_{\text{setup}}} \\ &= 1 + \frac{\frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}}}{\frac{2(U-1) \cdot l_c + 4 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{W+\frac{W}{2}+1}{v_{\text{pick}}} + 2 \cdot t_{\text{setup}}}.\end{aligned}$$

From $C \to \infty$ follows that

$$\frac{\mathcal{A}_{4.1}(\mathcal{I})}{\text{OPT}(\mathcal{I})} \overset{C \to \infty}{\longrightarrow} 2.$$

Selection Rule SHORT

Obviously, for the previous instance the selection rule SHORT does not lead to a competitive ratio of 2. For SHORT we consider the following example. At time 0, let two customer orders be available. Customer order i_4 requires $W-1$ items stored in cells of the first aisle, including one item located in the last cell of the first aisle. Customer order i_5 requires two items stored in the first cell of the leftmost aisle. The algorithm forms two batches, of which one includes customer order i_4 with service time

$$st_{i_4} = \frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{W-1}{v_{\text{pick}}} + t_{\text{setup}},$$

and of which the other includes customer order i_5 with processing time

$$st_{i_5} = \frac{2 \cdot l_{cw}}{v_{\text{travel}}} + \frac{2}{v_{\text{pick}}} + t_{\text{setup}}.$$

According to selection rule SHORT, the batch including customer order i_5 is started first. At the completion time of the batch containing i_5, a customer order identical to i_5 is available and is processed next. This step is repeated $k-1$ times where $k := \min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\}$. Algorithm 4.1 using SHORT will sequentially process each customer order which is identical to i_5 in a single batch, followed by the batch containing customer order i_4. In total, this leads to

$$\begin{aligned}\mathcal{A}_{4.1}(\mathcal{I}) &= k \cdot st_{i_5} + st_{i_4} \\ &= \min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\}(\frac{2 \cdot l_{cw}}{v_{\text{travel}}} + \frac{2}{v_{\text{pick}}} + t_{\text{setup}}) + \frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{W-1}{v_{\text{pick}}} + t_{\text{setup}}.\end{aligned}$$

An optimal off-line algorithm releases the batch including i_4 and then a batch j containing i_5 and the $k-1$ customer orders identical to i_5. The definition of k ensures that this second batch does not violate the capacity restriction. Therefore, we have

$$\begin{aligned}\text{OPT}(\mathcal{I}) &= st_{i_4} + st_j \\ &= \frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{W-1}{v_{\text{pick}}} + t_{\text{setup}} + \frac{2 \cdot l_{cw}}{v_{\text{travel}}} + \frac{2 \min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\}}{v_{\text{pick}}} + t_{\text{setup}}.\end{aligned}$$

For the competitive ratio the following expression can be calculated

$$\begin{aligned}\frac{\mathcal{A}_{4.1}(\mathcal{I})}{\text{OPT}(\mathcal{I})} &= \frac{\min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\}(\frac{2 \cdot l_{cw}}{v_{\text{travel}}} + \frac{2}{v_{\text{pick}}} + t_{\text{setup}}) + \frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{W-1}{v_{\text{pick}}} + t_{\text{setup}}}{\frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}} + \frac{W-1}{v_{\text{pick}}} + t_{\text{setup}} + \frac{2 \cdot l_{cw}}{v_{\text{travel}}} + \frac{2 \min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\}}{v_{\text{pick}}} + t_{\text{setup}}} \\ &= 1 + \frac{(\min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\} - 1)(\frac{2 \cdot l_{cw}}{v_{\text{travel}}} + t_{\text{setup}})}{\frac{2 \cdot (C+1) \cdot l_{cw}}{v_{\text{travel}}} + \frac{W-1+2 \min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\}}{v_{\text{pick}}} + 2 \cdot t_{\text{setup}}}.\end{aligned}$$

Additionally to the warehouse assumptions, we assume that $v_{\text{pick}} \to \infty$, $t_{\text{setup}} = 0$, and $W = 2 \cdot C$. In this case, we have

$$\begin{aligned}\frac{\mathcal{A}_{4.1}(\mathcal{I})}{\text{OPT}(\mathcal{I})} &= 1 + \frac{(\min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\} - 1)(\frac{2 \cdot l_{cw}}{v_{\text{travel}}})}{\frac{2 \cdot (C+1) \cdot l_{cw}}{v_{\text{travel}}}} = 1 + \frac{(\min\{\frac{W}{2}, \lfloor \frac{st_{i_4}}{st_{i_5}} \rfloor\} - 1)}{C+1} \\ &= 1 + \frac{(\min\{\frac{W}{2}, \lfloor \frac{\frac{2 \cdot C \cdot l_{cw}}{v_{\text{travel}}}}{\frac{2 \cdot l_{cw}}{v_{\text{travel}}}} \rfloor\} - 1)}{C+1} = 1 + \frac{\min\{\frac{2 \cdot C}{2}, C\} - 1}{C+1} \\ &= 1 + \frac{C-1}{C+1}.\end{aligned}$$

For $C \to \infty$ we obtain

$$\frac{\mathcal{A}_{4.1}(\mathcal{I})}{\text{OPT}(\mathcal{I})} \xrightarrow{C \to \infty} 2.$$

Summarizing, Algorithm 4.1 combined with all batching strategies and all selection rules does not have a better competitive ratio than 2.

5.3 Upper Bound

It remains to show that an algorithm exists whose competitive ratio is at most 2, independent of the kind of warehouse. In the following the proof of Zhang et al. (2001) for the problem of minimizing the makespan on a single bounded batch processing machine is adopted to the On-line Order Batching Problem. For this proof it is necessary that an optimal batching strategy $\mathcal{H}_{\text{opt}}$ is used, i.e. the algorithm obtains an optimal solution for the (off-line) Order Batching Problem. Let $\mathcal{A}^{\star}_{4.1}$ be the Algorithm 4.1 using $\mathcal{H}_{\text{opt}}$ and an arbitrary selection rule. Furthermore, let $\mathcal{A}^{\star}_{4.1}(\mathcal{I})$ denote the completion time of the latest batch, where $\mathcal{I}$ is an arbitrary problem instance. In this instance the arrival time of the last customer order is r_n (for the sake of simplicity it is assumed that exactly one customer order has the arrival time r_n). In order to prove that the competitive ratio of $\mathcal{A}^{\star}_{4.1}$ is at least 2, we distinguish between the following three cases.

- *All customer orders, which have arrived before customer order n, are already completed before r_n:* The described algorithm is an optimal one.

- *There are unprocessed customer orders and the order picker is idle:* In this case the solution of the batching problem at r_{n-1} leads to a single batch. According to the algorithm, the last batch j' includes all open customer orders without customer order n at time r_n and has the service time $st_{j'}$. The batch j' would be started at $2r_{i'} + st_{i'} - st_{j'}$ where i' is a customer order assigned to batch j' with the longest single service time $st_{i'}$. Since the order picker is idle, $2r_{i'} + st_{i'} - st_{j'}$ must be greater than r_n. Since the batching heuristic is optimal, $\mathcal{A}^{\star}_{4.1}$ will lead to a solution which is at least as good as a solution which will release j' followed by a seperated batch for customer order n. For $\mathcal{A}^{\star}(\mathcal{I})_{4.1}$ one obtains:

  $$\mathcal{A}^{\star}_{4.1}(\mathcal{I}) \leq r_n + st_{j'} + st_n < 2r_{i'} + st_{i'} - st_{j'} + st_{j'} + st_n = 2r_{i'} + st_{i'} + st_n$$

  If $st_{i'} \geq st_n$ holds, we can estimate the upper bound for $\mathcal{A}^{\star}_{4.1}(\mathcal{I})$ as follows:

  $$\mathcal{A}^{\star}_{4.1}(\mathcal{I}) < 2r_{i'} + st_{i'} + st_{i'} = 2(r_{i'} + st_{i'}) \leq 2\text{OPT}(\mathcal{I}).$$

  Otherwise ($st_{i'} < st_n$) we obtain

  $$\mathcal{A}^{\star}_{4.1}(\mathcal{I}) < 2r_{i'} + st_n + st_n \leq 2r_n + 2st_n = 2(r_n + st_n) \leq 2\text{OPT}(\mathcal{I}).$$

- *There are unprocessed customer orders and the order picker is not idle:* Let $\tilde{j}$ be the batch which is beeing processed while customer order n arrives. Let $s_{\tilde{j}}$ be the start time and $f_{\tilde{j}}$ be the completion time of $\tilde{j}$, therefore $s_{\tilde{j}} < r_n \leq f_{\tilde{j}}$. Let r be the earliest arrival time in the time interval $(s_{\tilde{j}}, f_{\tilde{j}}]$, i.e. $r = \min_i\{r_i | s_{\tilde{j}} < r_i \leq f_{\tilde{j}}\}$ and $A(r)$ be the set of the customer orders arriving after r, more formal $A(r) := \{i | r_i > r\}$. Let $st^{\star}_{A(r)}$ and $st^{\star}_{P(s_{\tilde{j}})}$ be the total service time of $A(r)$ and the set of open customer orders $P(s_{\tilde{j}})$ at time $s_{\tilde{j}}$, respectively. Furthermore, the inequalities $\text{OPT}(\mathcal{I}) \geq r + st^{\star}_{A(r)}$ and $\text{OPT}(\mathcal{I}) \geq st^{\star}_{P(s)}$ hold. Let $\tilde{I}$ be the set of customer orders assigned to batch $\tilde{j}$. All customer orders of $\tilde{I}$ must be available before $\tilde{j}$ is started and, therefore, $\tilde{I} \subset P(s_{\tilde{j}})$. The solution of the on-line

algorithm is $\mathcal{A}^{\star}_{4.1}(\mathcal{I}) = f_{\tilde{j}} + st^{\star}_{A(r)\cup\{P(s_{\tilde{j}})\setminus\tilde{I}\}}$, where $st^{\star}_{A(r)\cup\{P(s_{\tilde{j}})\setminus\tilde{I}\}}$ is the service time for a set of batches for the customer orders in $A(r) \cup \{P(s_{\tilde{j}}) \setminus \tilde{I}\}$. Since the algorithm uses an optimal batching strategy, $st^{\star}_{A(r)\cup\{P(s_{\tilde{j}})\setminus\tilde{I}\}}$ is the optimal service time for the customer orders in $A(r) \cup \{P(s_{\tilde{j}}) \setminus \tilde{I}\}$. Let further $st^{\star}_{P(s_{\tilde{j}})\setminus\tilde{I}}$ be the optimal service time for $P(s) \setminus \tilde{I}$.

Then,

$$st^{\star}_{A(r)\cup\{P(s_{\tilde{j}})\setminus\tilde{I}\}} \leq st^{\star}_{P(s_{\tilde{j}})\setminus\tilde{I}} + st^{\star}_{A(r)}$$

is valid. For the estimation of the upper bound, we obtain

$$\begin{aligned}\mathcal{A}^{\star}_{4.1}(\mathcal{I}) = f_{\tilde{j}} + st^{\star}_{A(r)\cup\{P(s)\setminus\tilde{I}\}} &\leq f_{\tilde{j}} + st^{\star}_{P(s_{\tilde{j}})\setminus\tilde{I}} + st^{\star}_{A(r)} \\ = s_{\tilde{j}} + st^{\star}_{P(s_{\tilde{j}})} + st^{\star}_{A(r)} &\leq r + st^{\star}_{P(s)} + st^{\star}_{A(r)} \leq 2\mathrm{OPT}(\mathcal{I}).\end{aligned}$$

To summarizes, $\mathcal{A}^{\star}_{4.1}$ has at most the competitive factor 2. Combined with the consideration above, that the competitive factor is at least 2, we can conclude that this bound is tight if an optimal batching heuristic is used.

6 Purpose and Design of the Numerical Experiments

6.1 Purpose

In order to determine the solution quality of the described algorithms, an extensive series of numerical experiments has been carried out. A variety of problem classes has been considered and the behavior of the suggested approaches was simulated. The experiments aims at investigating which selection rule should be combined with a particular batching strategy in order to generate a short completion time of all customer orders. Furthermore, we explore and compare the solution quality of the different batching heuristics.

6.2 Warehouse Parameters

In our experiments a single-block warehouse with two cross aisles, one in the front and one in the back of the picking area, is assumed. This layout type (cf. Figure 5.1) has been used frequently in experiments described in the literature (Gademann and van de Velde, 2005; Henn et al., 2009). We specify the parameters in the following way: The picking area consists of 900 storage locations, where a different article has been assigned to each storage location. The storage locations are arranged into 10 aisles ($U = 10$) with 90 storage locations each (45 cells on both sides of an aisle, i.e. $C = 45$). The aisles are numbered from 1 to 10; aisle no. 1 is the leftmost aisle and aisle no. 10 the rightmost one. Each cell has a width of one length unit (LU) ($l_{cw} = 1$) and the center-to-center distance between two aisles amounts to 5 LU ($l_c = 5$). The depot is 1.5 LU away from the first storage location of the leftmost aisle and the distance between the front cross aisle and the depot amounts to 0.5 LU. We further assume that an order picker walks 10 length units

in 30 seconds and he/she needs 10 seconds to search and collect an article from a storage location. This results in a travel velocity of $v_{\text{travel}} = 48[\frac{\text{LU}}{\text{min}}]$ and a pick velocity of $v_{\text{pick}} = 6[\frac{\text{items}}{\text{min}}]$. For each tour a setup time t_{setup} of 3 minutes is needed. We consider a *class-based storage assignment*, the articles are grouped into three classes A, B, and C according to their expected demand frequency. A contains articles with high, B with medium and C with low demand frequency. Articles of class A are only stored in aisle no. 1, articles of B in the aisles no. 2, no. 3, and no. 4., and articles of class C in the remaining six aisles. Furthermore, it is assumed that 52 percent of the demanded articles belong to articles in class A, 36 percent to articles in B and 12 percent to articles in C. Within a class, the location of an article is determined randomly. All parameters are summarized in Table 6.1.

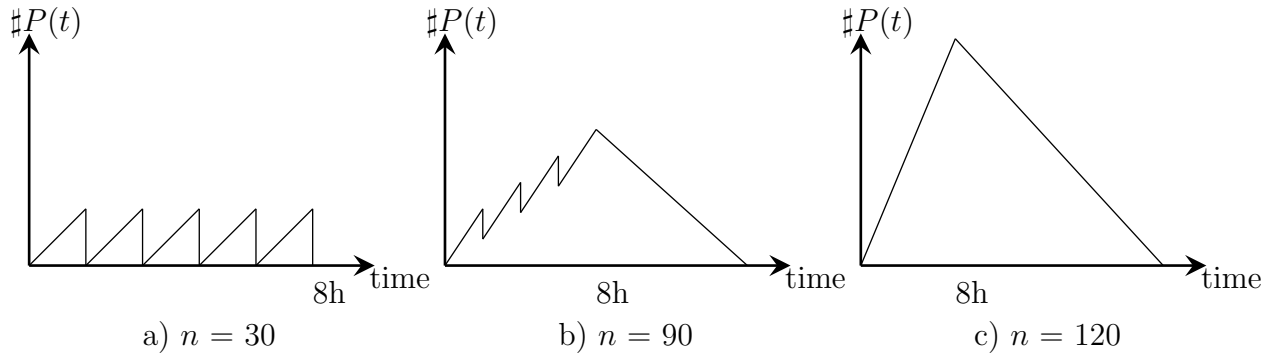
Attributes	**Values**
number of aisles (U):	10
number of cells on each side of an aisle:	45
width of a storage location (l_{cw}) [LU]:	1
center-to-center-distance between two aisles (l_c) [LU]:	5
distance between depot and front cross aisle:	0.5
pick velocity v_{pick} $[\frac{\text{articles}}{\text{min}}]$:	6
travel velocity v_{travel} $[\frac{\text{LU}}{\text{min}}]$:	48
setup time t_{setup} [min]:	3

Table 6.1: Warehouse layout and order picker parameters

6.3 Problem Classes

To analyze the quality of the proposed algorithms we vary several problem parameters. For the capacity of the picking device W we assume two different values, namely 45 and 75 items. For the routing strategy, the S-Shape heuristic and the Largest-Gap heuristic are used. For a customer order we choose the quantity of items uniformly distributed in $\{5, \ldots, 25\}$, resulting in 3 or 5 customer orders per batch on average, in accordance with the above defined capacities of the picking device. For the total number of customer orders n we consider 30, 60, 90 and 120. The customer orders should arrive within a planning period of eight hours. The *interarrival times* – the time between the arrival of customer order i and customer order $i + 1$ – are exponentially distributed with the *arrival rate* λ. Let $X(t)$ be the number of incoming customer orders in the time interval $[0, t]$. The stochastic process $\{X(t)|t \geq 0\}$ is called *arrival process*. In the case of exponentially distributed interarrival times $E[X(t)] = \lambda \cdot t$ holds. In our numerical experiments we choose the arrival rate λ in a way that the expectation $E[X(t)]$ is equal to n for $t = 8$[h]. In summary, we use the following values for λ: for $n = 30$: $\lambda = 0.08625$, for $n = 60$: $\lambda = 0.125$, for $n = 90$: $\lambda = 0.1875$, and for $n = 120$: $\lambda = 0.25$.

Chew and Tang (1999) provide the expected travel time related to a batch containing k articles,



a) $n = 30$ b) $n = 90$ c) $n = 120$

Figure 6.1: Number of open orders for different numbers of customer orders

when the routes are determined by means of the S-Shape heuristic. In order to determine the service time st_i of a single customer order i we use a simplified version of this expression and obtain

$$E[st_i] \approx \frac{l_a}{v_{\text{travel}}}(U - \sum_{u=1}^{U}(1-p_u)^k + \frac{1}{2}) + \frac{2l_c}{v_{\text{travel}}}(U - \sum_{u=1}^{U-1}(\sum_{r=1}^{u} p_r)^k) + \frac{k}{v_{\text{pick}}} + t_{\text{setup}}, \qquad (6.1)$$

where p_u ($u \in \{1, \ldots, U\}$) is the probability that an items is picked in aisle u. According to the classed based storage policy we obtain an expected travel time of 12.49 min for an customer order with 15 items. If we assume that each customer order is assigned to a single batch, 48 customer orders could be processed within eight hours. The different values for n lead to the following situations: For $n = 30$ all customer orders can be processed within the eight hours and the order picker can process each open customer order in a single tour. During this tour new customer orders arrive. In problem classes where n is greater than 30, customer orders arrive faster than the order picker can process the open orders. Consequently, the number of open orders increases till the last customer order has arrived. After that the number of open orders decreases. Figure 6.1 shows the number of open orders during the eight hours for different n.
Combination of the test parameters described above leads to 16 problem classes, which are summarized in Table 6.2. For each problem class 50 instances have been generated, which provided 800 instances in total.

Attribute	**Values**
capacity of a picking device W	45, 75 items
routing strategy	S-Shape, Largest Gap
number of customer orders n	30, 60, 90, 120

Table 6.2: Problem classes

6.4 Algorithm Parameters

For the experiments the parameters of the algorithms have to be specified. The topic of this paper is a real time problem. Therefore, the fast generation of solutions is an essential requirement for

each solution approach. Since the point in time when the order picker becomes available again is known at the start time of a batch, the time interval between start time and completion time can be used to generate solutions. Since new customer orders, which should be considered also, may arrive during this time interval, a limitation of the computing time is necessary. Therefore, we restrict the termination condition in ILS (Henn et al., 2009) to one minute. The other parameters are chosen as described in Table 6.3.

Attributes	**Values**
termination condition [min]:	1.00
rearrangement factor θ:	0.30
threshold factor μ:	0.05
time interval $t_{\text{incumbent}}$[min]:	0.20

Table 6.3: Parameters for batching heuristic ILS

6.5 Implementation and Hardware

The computations for all 800 instances have been carried out on a Pentium processor with 2.21 GHz and 2.0 GB RAM. The algorithms have been implemented with C++ using the DEV Compiler Version 4.9.9.2.

7 Results of the Experiments

7.1 Outline

In this section the results of the numerical experiments are presented, differentiated with respect to the routing strategies S-Shape and Largest Gap. The tables 7.1 and 7.3 depict the average completion time of all customer orders provided by the algorithms for the problem class. Additionally, the tables 7.2 and 7.4 contain the average turnover times of a customer order for different problem classes. The first column in each table describes the problem class, represented by "total number of customer orders / capacity of the picking device in number of items" (n/W). The entries in the other columns show the results of Algorithm 4.1 combined with the batching heuristics FCFS, C&W(ii), and ILS and the selection rules FIRST, SHORT, LONG, and SAV. The best average value generated by a selection rule for the application of a particular batching heuristic is highlighted bold. The results are analyzed as follows: For each routing strategy we compare the impact of the selection rules for a particular batching method. Therefore, the selection rule, which leads to the best results, serves as benchmark. Afterwards, we evaluate the solution quality of the different batching strategies combined with the corresponding selection rule, which leads to the best results.

7.2 S-Shape-Routing

Class	Algorithm 4.1 with											
	FCFS and				C&W(ii) and				ILS and			
	FIRST	SHORT	LONG	SAV	FIRST	SHORT	LONG	SAV	FIRST	SHORT	LONG	SAV
30/45	**467**	488	**467**	**467**	471	484	**465**	**465**	469	485	467	**466**
30/75	**457**	482	**457**	**457**	472	483	**457**	**457**	**456**	485	457	458
60/45	519	551	**518**	**518**	518	541	**511**	**511**	511	531	511	**510**
60/75	**499**	526	**499**	**499**	505	521	**502**	**502**	**501**	520	**501**	**501**
90/45	651	722	**638**	643	605	682	**593**	598	579	633	**575**	576
90/75	520	593	519	**518**	520	578	513	**512**	510	549	**508**	**508**
120/45	856	930	**840**	847	766	851	**758**	762	731	790	732	**730**
120/45	634	733	**627**	630	613	701	**602**	**602**	589	647	586	**585**

Table 7.1: Average completion time of the last batch in minutes for S-Shape-Routing

If customer orders are grouped by means of the FCFS rule, the selection rule LONG obtains the smallest completion times for 7 out of 8 problem classes on average. SAV obtains the best objective function value for 5, FIRST for 3, and SHORT for no problem class. The results provided by the selection rules LONG, FIRST and SAV are nearly identical for 30, 60 customer orders, as well as and for the problem class with 90 customer orders and a capacity of 75 items. For the remaining problem classes the results obtained for LONG and SAV differ by at least 7 min. The deviation of the results obtained by LONG and FIRST amounts to 14 min for the problem class with $n = 90$ and $W = 45$; for the problem classes with 120 customer orders the difference amounts to 16 and 7 min. By the application of SHORT the completion times of all customer orders are significantly larger than those generated by LONG: around 30 min for problem classes with 30 and 60 customer orders, more than 70 and less than 90 min for problem classes with 90 customer orders, and more than 90 and less than 106 min for $n = 120$.

If the batching heuristic C&W(ii) is used, LONG and SAV provide the best results for 7 and 6 problem classes, respectively. The remaining selection rules do not find any best (average) result. Applying FIRST all customer orders are completed up to 15 min later than in case of LONG. The deviation between the results of SHORT and LONG amounts to 20 min for $n = 30$, to 30 min for $n = 60$ and a small capacity, and to 19 min for $n = 60$ and a large capacity. By the application of SHORT all customer orders for problem classes with 90 customer orders are completed 89 and 65 min later than by the application of LONG. For 120 customer orders the deviation between SHORT and LONG amounts to more than 90 min.

By the application of ILS, SAV leads to the best results for 6 problem classes, LONG for 3, FIRST for 2 problem classes. SHORT does again not obtain a best result at all. The behavior of the selection rules FIRST, LONG, and SAV is almost identical in terms of the completion time of all customer orders. The differences of the solutions provided by SHORT as compared to the ones provided by SAV are significantly larger: approximately 20 min for 30 and 60 customer orders, more than 40 min for 90 customer orders, and around 60 min for $n = 120$.

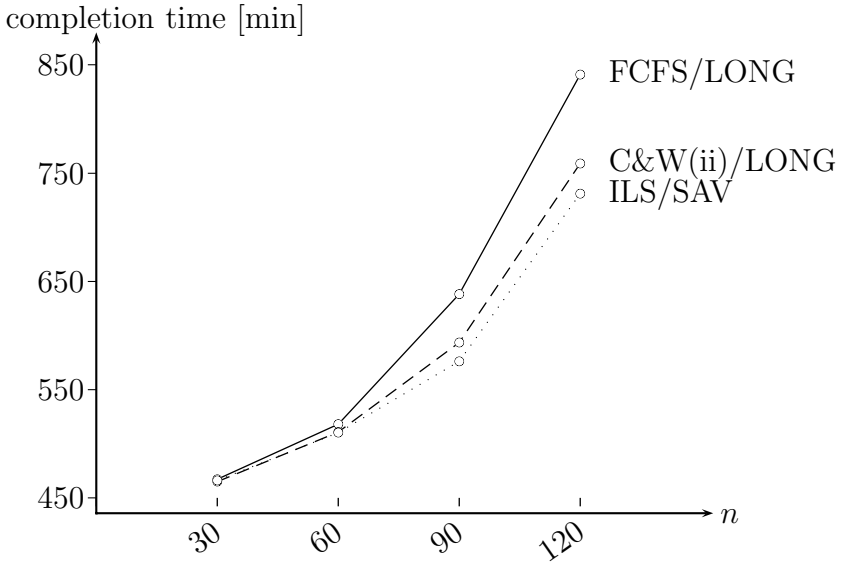


Figure 7.1: Completion time of all orders for $W = 45$ and S-Shape

The results of the different selection rules are almost identical for small numbers of customer orders. Due to the fact that, more than 30 customer orders can be processed during the eight hours, the number of open customer orders is small during this time. At each decision point the number of generated batches is small. Therefore, the selection rules determine identical batches, which are released. Also, in the case of a higher capacity of the picking device the number of available batches at a specific decision point is smaller than in the case of a small capacity. Therefore, the different selection rules release identical batches for the larger capacity. The selection rule SHORT tends to select batches with a small capacity utilization, since a small number of items in a batch results in a small service time. As a consequence, in the case of SHORT the number of released batches is larger than in the remaining selection rules, which results in a larger completion time of all customer orders.

Comparing the results of FCFS/LONG (the batching heuristic FCFS in combination with the best selection rule LONG) to the results obtained by C&W(ii)/LONG, the latter outperforms the first one for large problem classes, while the results are very similar for the classes with 30 and 60 customer orders, and for the problem class with 90 customer orders and a capacity of 75 items. In the remaining problem classes the difference amounts to 45 min ($n = 90/W = 45$), 82 min (120/45) and 25 min (120/75). In total, we conclude, that the difference increases with a smaller capacity of the picking device, as well as with a larger number of customer orders. If ILS/SAV is compared to C&W(ii)/LONG, the use of ILS is preferred to C&W(ii). The results of both batching heuristics are similar for the problem classes with 30 and 60 customer orders and the problem class with 90 customer orders and a capacity of 75 items. For the problem class with 90 customer orders combined with a capacity of 45 items and in the problem class with 120 customer orders combined with a capacity of 75 items a difference of 17 min can be observed. In the class with 120 customer orders and a capacity of 45 items the results even differ by 28 min. Figure 7.1 visualizes the development of the solution quality for the three combinations by

Class	Algorithm 4.1 with											
	FCFS and				C&W(ii) and				ILS and			
	FIRST	SHORT	LONG	SAV	FIRST	SHORT	LONG	SAV	FIRST	SHORT	LONG	SAV
30/45	**44**	55	**44**	**44**	**47**	50	**47**	**47**	**46**	50	**46**	**46**
30/75	**57**	72	**57**	**57**	63	67	**60**	**60**	**59**	69	60	**59**
60/45	45	66	45	**44**	45	57	44	**43**	42	52	42	**41**
60/75	**46**	63	**46**	**46**	51	56	**50**	**50**	48	56	**48**	**48**
90/45	113	161	106	**96**	92	129	94	**79**	81	102	84	**70**
90/75	55	102	54	**53**	57	84	55	**53**	51	70	51	**50**
120/45	207	260	195	**175**	161	203	173	**142**	148	166	159	**128**
120/75	101	170	98	**93**	94	138	95	**84**	82	109	85	**76**

Table 7.2: Average turnover time per customer order in minutes for S-Shape-Routing

increasing n and for $W = 45$.

In summary, the impact of the batching strategy is neglectable for small numbers of customer orders. For large numbers of customer orders the application of ILS/SAV can save up to 90 min in comparison to FCFS/LONG. By the application of ILS and C&W(ii) the results obtained by the used selection rules do not differ as large than in the case where batching is done by means of the FCFS rule. A more sophisticated batching heuristic generates more balanced batches (in terms of service time), as well as a smaller number of batches.

If – as an alternative objective function – the minimization of the average turnover times of a customer order is chosen, slightly different results for the best combination of batching heuristic and selection rule can be identified. As described in Table 7.2 SAV outperforms the other selection rules. Comparing the batching heuristics, ILS also leads to significantly smaller turnover times than FCFS.

7.3 Largest Gap-Routing

Class	Algorithm 4.1 with											
	FCFS and				C&W(ii) and				ILS and			
	FIRST	SHORT	LONG	SAV	FIRST	SHORT	LONG	SAV	FIRST	SHORT	LONG	SAV
30/45	466	485	466	**465**	468	482	**465**	466	**465**	482	468	468
30/75	**458**	482	**458**	**458**	466	485	**451**	**451**	452	484	**451**	**451**
60/45	514	539	**513**	514	513	535	**509**	510	510	528	**507**	508
60/75	**500**	523	**500**	**500**	506	523	**500**	**500**	**498**	521	500	499
90/45	630	690	**618**	622	593	660	**582**	586	573	626	**569**	572
90/75	525	586	524	**523**	525	577	**515**	**515**	513	555	**511**	512
120/45	827	893	**813**	821	755	834	**743**	750	727	786	**725**	727
120/75	646	731	**641**	643	622	704	**610**	614	601	661	**596**	597

Table 7.3: Average completion time of the last batch in minutes for Largest Gap routing

In the problem classes where routing is done by the Largest Gap heuristic, very similar results like in the case of S-Shape-Routing can be observed. For FCFS and C&W(ii) the selection rule LONG leads again to the best results on average. Only for ILS a different selection rule, namely LONG, generates the best results for Largest Gap-Routing than in the case of S-Shape-Routing. The relations of the results provided by the application of the different selection rules and a particular batching heuristic are very similar to the case of S-Shape-Routing. Comparing the results of the different batching heuristic, the combination ILS/LONG outperforms the other strategies. Figure 7.2 visualizes the differences if the capacity of the picking device amounts to 45 items.

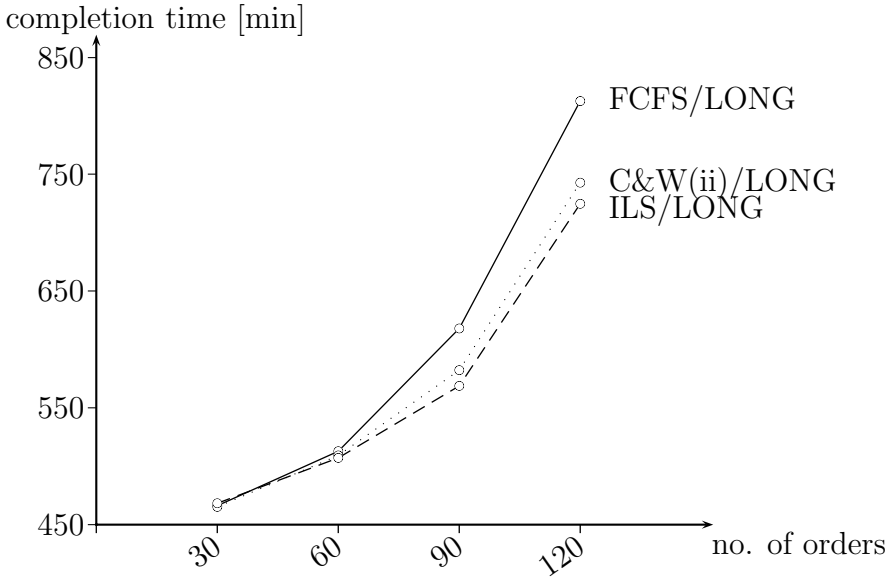


Figure 7.2: Completion time of all orders for $W = 45$ and Largest Gap-Routing

From the average turnover times depicted in Table 7.4 it can be concluded that the application of SAV leads in every problem class to the smallest turnover times of a customer order. Similar to the results of S-Shape routing ILS provides the best average turnover times.

Class	**Algorithm 4.1 with**											
	FCFS and				**C&W(ii) and**				**ILS and**			
	FIRST	SHORT	LONG	SAV	FIRST	SHORT	LONG	SAV	FIRST	SHORT	LONG	SAV
30/45	**43**	53	44	**43**	47	48	**46**	**46**	**44**	49	46	46
30/75	**58**	72	**58**	**58**	62	66	**59**	**59**	**59**	68	**59**	**59**
60/45	43	59	43	**41**	44	52	43	**41**	42	50	42	**40**
60/75	**47**	61	**47**	**47**	51	55	**50**	**50**	49	55	**49**	**49**
90/45	102	141	97	**88**	86	115	87	**74**	77	98	78	**68**
90/75	58	95	57	**56**	60	81	58	**55**	53	73	53	**52**
120/45	192	239	183	**163**	156	193	167	**134**	147	168	153	**127**
120/75	107	163	104	**98**	98	133	102	**88**	88	116	90	**81**

Table 7.4: Average turnover time per customer order in minutes for Largest Gap routing

8 Conclusions and Outlook

This paper deals with the on-line variant of the Order Batching Problem, one of the three main planning problems in a manual picker-to-parts warehouse. The problem is to transform customer orders, arriving over time, into picking orders such that the completion time of all customer orders is minimized. Existing methods for the corresponding off-line Order Batching Problem, namely First-Come-First-Served, C&W(ii), and Iterated Local Search have been modified for this on-line problem. By means of a competitive analysis it is shown that the general principle of this algorithm is 2-competitive in combination with an optimal batching algorithm. The analysis also showed that every on-line algorithm for this problem is at least 2-competitive. Extensive numerical experiments have been carried out to evaluate which heuristic leads to the best results. The selection rules LONG and SAV provide the best completion times independent of the choice of the batching heuristic. Since ILS provides significantly better results than FCFS and C&W(ii) it is recommended that batching is done as good as possible while the search for an appropriate selection rule is less significant. Furthermore, if the turnover time of a customer order is important, the selection rule SAV should be preferred.
For further research we suggest to investigate the impact of different warehouse layouts (two block warehouses, non standard warehouses etc.) and other kinds of storage policies. An other research object should be to consider due dates.

References

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

Caron, F., G. Marchet, and A. Perego (1998). Routing Policies and COI-Based Storage Policies in Picker-to-Part Systems. *International Journal of Production Research 36*(3), 713–732.

Chen, M.-C. and H.-P. Wu (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. and L. Tang (1999). Travel Time Analysis for General Item Location Assignment in a Rectangular Warehouse. *European Journal of Operational Research 112*(3), 582–597.

Clarke, G. and J. Wright (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research 12*(4), 568–581.

de Koster, M., E. van der Poort, and M. Wolters (1999). Efficient Orderbatching Methods in Warehouses. *International Journal of Production Research 37*(7), 1479–1504.

de Koster, R., T. Le-Duc, and K. Roodbergen (2007). Design and Control of Warehouse Order Picking: A Literature Review. *European Journal of Operational Research 182*(2), 481–501.

de Koster, R., K. Roodbergen, and R. van Voorden (1999). Reduction of Walking Time in the Distribution Center of De Bijenkorf. In M. Speranza and P. Stähly (Eds.), *New Trends in Distribution Logistics*, pp. 215–234. Berlin: Springer.

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

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

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

Elsayed, E. and O. Unal (1989). Order Batching Algorithms and Travel-time Estimation for Automated Storage/Retrieval Systems. *International Journal of Production Research 27*(7), 1097–1114.

Fiat, A. and G. Woeginger (1998). Competitive analysis of algorithms. In A. Fiat and G. Woeginger (Eds.), *Online algorithms: The state of the art*, Volume 1442 of *Lecture Notes in Computer Science*, pp. 1–12. Springer.

Gademann, A., J. van den Berg, and H. Van der Hoff (2001). An Order Batching Algorithm for Wave Picking in a Parallel-Aisle Warehouse. *IIE Transactions 33*(5), 385–398.

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

Gibson, D. and G. Sharp (1992). Order Batching Procedures. *European Journal of Operational Research 58*(1), 57–67.

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

Henn, S., S. Koch, K. Doerner, C. Strauss, and G. Wäscher (2009). Metaheuristics for the Order Batching Problem in Manual Order Picking Systems. Working Paper 20/2009, Faculty of Economics and Management, Otto-von-Guericke-University Magdeburg.

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

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

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

Le-Duc, T. (2005). *Design and Control of Efficient Order Picking Processes*. Ph. D. thesis, Erasmus Resarch Institute of Management, RSM Erasmus University Rotterdam.

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

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

Ratliff, H. and A. Rosenthal (1983). Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem. *Operations Research 31*(2), 507–521.

Sleator, D. and R. Tarjan (1985). Amortized Efficiency of List Update and Paging Rules. *Communications of the ACM 28*(2), 202–208.

Tang, L. and E.-P. Chew (1997). Order Picking Systems: Batching and Storage Assignment Strategies. *Computers & Industrial Engineering 33*(3-4), 817–820.

Tsai, C.-Y., J. Liou, and T.-M. Huang (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. and R. de Koster (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 H. Dyckhoff, R. Lackes, and J. Reese (Eds.), *Supply Chain Management and Reverse Logistics*, pp. 323–347. Berlin et al.: Springer.

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

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

Zhang, G., X. Cai, C.-Y. Lee, and C. Wong (2001). On-line Algorithms for Minimizing Makespan on Batch Processing Machines. *Naval Research Logistics 48*(3), 241–258.