A Simulated Annealing Approach for the Joint Order Batching and Order Picker Routing Problem with Weight Restrictions

Eric H. Grosse
Christoph H. Glock
Technische Universität Darmstadt
(grosse@bwl.tu-darmstadt.de)
(glock@bwl.tu-darmstadt.de)

Rafael Ballester-Ripoll
University of Zurich
(rballester@ifi.uzh.ch)

Volume 20*, Number* 2
September 2014*, pp.* 101-119

This paper studies the joint order batching and order picker routing problem in conventional multi parallel-aisle picker-to-part order picking systems. It complements prior publications by considering capacity constraint that is formulated as a function of total item weight, instead of item count. A mathematical model is formulated and a simulated annealing algorithm is developed to batch orders and to determine pick tours. The intention of the paper is to provide a more realistic model and to improve classical batching and routing heuristics. It thereby pays special attention to the practical applicability of the model. The proposed methods are compared and evaluated in an extensive numerical study, and it is shown that the developed approach leads to an improved solution for the joint order batching and order picker routing problem, as compared to classical heuristics for this problem.

Keywords: Order Picking, Order Batching, Warehouse, Simulated Annealing, Picker-to-Part System

1. Introduction

Order picking, the manual retrieval of items from storage locations to satisfy customer orders, has been identified as one of the most labor- and time-intensive internal logistics processes (Frazelle 2002; Tompkins et al. 2010). The reason for the high cost impact of order picking is that orders are picked manually in many cases. Human order pickers are often difficult to replace because of their flexibility and their cognitive and motor skills that are hard to imitate by machines. As a result, order picking is a labor- and time-intensive task, and improving order picking efficiency and reducing warehouse operating costs is essential for many companies (Le-Duc and de Koster 2005).A primary objective of warehouse optimization is the reduction of travel distance and thus travel time, where the latter refers to the time the order picker requires for traveling from the depot to the shelves of the warehouse and back. Some researchers estimated that travel time is responsible for up to 55% of the total time the order picker requires to complete an order (Tompkins et al. 2010; de Koster et al. 2007). Travel time is influenced by order picking routing policies, which determine the sequence of item retrieval in the warehouse

and the sequence in which aisles are traversed. The problem becomes more complex if the carrying capacity of the order picker is limited, which could be the result ofthe picker's physical capabilities or the loading capacity of the transportation equipment used, for example. In such a situation, large orders have to be split up into so-called batches, which are then picked successively. In the case of small orders, several orders with only a few items could be combined in a single batch to make a better use of the picker's capacity.

A closer look at the literature reveals that works that considered a limited carrying capacity of the order picker typically assumed that the maximum number of items an order picker is able to carry is restricted. It is obvious that this assumption does not reflect reality completely, as items that are stored in an order picking warehouse are often heterogeneous with different sizes and weights, which are item characteristics that could also limit the maximum number of products the order picker is able to carry(Grosse et al. in Press). To provide a more realistic model formulation, we extend prior models by assuming that the capacity of the order picker is limited in terms of the total weight he/she is able to carry. First, a model of a joint batching and order picker routing problem is formulated in this paper, and then a simulated annealing (SA) approach is developed to determine order batches and picker routes. The routing heuristics and the SA improvement strategy studied in this paper can be implemented in different types of warehouses in practice, and they promise improvements in order picking without requiring high capital investments or time.

The remainder of the paper is organized as follows: The next section reviews the relevant literature. Section 3 describes the layout and the configuration of the warehouse studied in this paper. Section 4 presents several batching heuristics and develops a SA approach for batch improvement. Section 5 reports the results of an extensive numerical experiment, and Section 6 concludes the paper. For the sake of brevity, the male gender is used to refer to individuals that could be male or female.

2. Literature Review

The literature on order picking can roughly be differentiated into four main research streams: works on storage assignment, on layout design, on routing, and on order batching. The aim of works in this area usually is to reduce the average distance the order picker needs to travel to complete a given set of customer orders. In the following, we give a brief overview of optimization problems that occur in the context of order picking, discuss relevant works on order batching and show how SA has been applied to warehouse management problems in the past. For a detailed review of the literature on order picking, the reader is referred to Gu et al. (2007) and de Koster et al. (2007).

2.1 Planning Problems in Order Picking

- The first planning problem that arises in order picking is the assignment of products to storage locations. If a random storage assignment is used, items arriving at the warehouse are assigned to a random open location in the warehouse. The closest open location policy, in turn, assigns an item to the next open location in the warehouse that is closest to the depot. It is clear that for both policies, the location of an item in the warehouse may change over time. A dedicated storage assignment, in contrast, assigns products to shelf locations based on item characteristics such as demand frequency or volume (Frazelle 2002), where item locations remain fixed once they have been determined, even

if the item is out of stock. Items with high demand frequencies, for example, could be assigned to locations close to the depot (Hausmann et al. 1976; Petersen et al. 2004; Bottani et al. 2012). Another option is to consider the effort for refilling shelves as well in assigning products to shelf locations. Thus, items that need to be refilled frequently should be stored close to the reserve area of the warehouse (Gu et al. 2010). The cube-per-order index (COI), which is the ratio of the space required by an item to its order frequency, is helpful in this case, as a low COI indicates that the item has to be restocked frequently. Items with a low COI should therefore be stored close to the depot or the reserve area (Malmborg 1995; Muppani and Adil 2008). Finally, also demand correlations could be used in assigning items to shelf locations (Bindi et al. 2009; Grosse and Glock 2012).

- The second planning problem, layout design, determines the aisle configuration of the warehouse, which includes the numbers of aisles and cross aisles as well as their dimensions (Petersen 2002; Roodbergen and Vis 2006;Roodbergen et al. 2008). The standard layout of rectangular shape with parallel aisles is very common in practice and has frequently been analyzed in the literature (Ratliff and Rosenthal 1983; Gibson and Sharp 1992; Petersen and Schmenner 1999;de Koster et al. 1999;Hwang et al. 2004;Bozer and Kile 2008; Henn et al. 2010). More recently, other layouts have been introduced, such as order picking areas with U-shaped or fishbone layouts (Glock and Grosse 2012; Çelik and Süral2013).
- Routing policies define the sequence in which the order picker retrieves items from the shelves of the warehouse, as well as the sequence in which the picker travels through the aisles of the warehouse. Ratliff and Rosenthal (1983), for example, formulated the order picker routing problem as a variant of the travelling salesman problem and calculated optimal routes. Rood bergen and de Koster (2001) developed a dynamic programming algorithm to find order picking tours of minimal length. In many scenarios, however, computing optimal tours is not possible due to the mathematical complexity of the optimization problem. For this reason, and to provide routing policies that can easily and flexibly be implemented in practice, researchers developed different routing heuristics in the past. Another reason for applying routing heuristics in practice is that optimal routes may seem illogical to the order pickers, as they do not necessarily follow an easy-to-remember pattern, which is why they may induce the order picker to deviate from the routes (Gademann and van de Velde 2005). The efficiency of routing heuristics has been evaluated frequently, for example by Goetschalckx and Ratliff (1988), Petersen and Schemer (1999), Hwang et al. (2004), Petersen and Aase (2004), and Theys et al.(2010). Routing heuristics that have often been studied in the literature are the following:
- Traversal Policy (or S-shape heuristic): The order picker traverses each aisle that contains at least one pick completely and then returns to the depot.
- Return Policy: The order picker enters each aisle that contains at least one ‘'pick and leaves the aisle on the side where he entered it. After all items have been picked, the order picker returns to the depot.
- Midpoint Policy: The warehouse is divided into two halves. The order picker first enters the side of the aisle that is located closest to the depot (the front side) to retrieve items located on this side of the aisle. After the front side of an aisle

has been covered, the order picker leaves the aisle on the side where he entered it. Once the front sides of all aisles have been covered, the picker travels to the back side of the warehouse and picks all items located on this side. After all items have been picked, the order picker returns to the depot.

- Largest Gap Policy: The order picker travels in an aisle only up to the largest gap between two picks within an aisle. This is similar to the midpoint policy, except that the largest gap between two picks is used as the reference point for turning around, and not the midpoint of the aisle. Again, the picker returns to the depot once all items have been picked.
- Composite Policy: The order picker combines any of the policies described above.

Caron et al. (1998) compared the return and traversal policies and suggested that the choice of a routing policy should depend on the number of picks per aisle.

Another planning problem in order picking is commonly referred to as order batching. Order batching for warehouse management derives from batching in production management, where production lots (or orders) are split up or consolidated to determine economic batch sizes that balance transportation, inventory and setup costs (cf., for example, Glock et al. in Press; Savino et al. 2010). In warehouse management, it is also necessary to generate batches if individual orders are too large or contain too few items, as compared to the order picker's carrying capacity. If orders are too large, they need to be split up into several sub-tours (batches) that are then picked successively (de Koster, et al. 1999; Gademann and van de Velde 2005). If orders only consist of few items, in contrast, combining several orders into a single batch can lead to efficiency improvements as it would be inefficient to fulfill orders with only low quantity of items in individual tours (Gademann and van de Velde, 2005). The reader may also refer to van den Berg (1999), de Koster et al. (2007), Gu et al. (2007) and Kulak et al. (2012) for an overview of the order batching problem. To solve the order batching problem, Gademann et al. (2001), for example, developed a branch-and-bound algorithm to solve medium-sized batching problems exactly. However, as the order batching problem is a variation of the classical vehicle routing problem and NP-hard if a batch contains more than two orders (Gademann and van de Velde, 2005), most authors developed heuristics for assigning items to batches. Gademann and van de Velde (2005) developed a branch-and-price algorithm that led to good results for a small number of orders. Gibson and Sharp (1992)developed several batching heuristics and combined them with experimental factors, such as item location, number of items per order or number of orders. They showed that combining their batching heuristic with a skewed item location may lead to a reduction in the average tour length of up to 44%. They assumed a pick device capacity of 50 items. De Koster et al. (1999) also developed heuristics for the order batching problem and assumed that at most eight items could be carried by the order picker. Their approach was sequential, i.e. order batches were generated first, and then two routing policies were applied. One batching heuristic they used was based on the savings algorithm (Clarke and Wright 1964),and it performed best for small capacity limits and if combined with largest gap routing. Another heuristic used a seed algorithm, and it led to good results for large numbers of orders and if it was combined with the S-shape routing policy. Hsu et al. (2005) developed a genetic algorithm and showed that this algorithm is robust and can be applied to any batch structure and warehouse layout. They assumed a pick device capacity of 100 to 500 items. A similar approach was used by Hennet al. (2010), who developed an iterated local search procedure and an ant colony

optimization algorithm. They fixed the capacity of the picking device to 30, 45, 60, and 75 articles. Bozer and Kile (2008) used the traversal routing policy to develop a mixed-integer programming model for the order batching problem. The capacity of the order picker was again formulated as a function of the item count. Chen and Wu (2005) also presented an integer programming approach and combined it with data mining. They fixed the picker capacity to 100, 200 and 500 items. In a recent study, Hsieh and Huang (2011) developed two batch construction heuristics, which they termed 'k-means batching' and 'self-organisation map batching'. Both heuristics used data mining and assumed a capacity constraint of 50 items. Hwang and Kim (2005)developed a batching algorithm based on a cluster analysis that was evaluated in combination with traversal, return and midpoint routing and under random storage assignment and a picker capacity of 24 items. They showed that the cluster analysis leads to good results when the order size exceeds 20 items. Parikh and Meller (2008) compared order batching and zoning, where zoning refers to a system where an order picker is responsible for picking items in a specific region (zone) of the warehouse. The authors used a picker capacity of 75 items. Cluster-based tabu search algorithms were used by Kulak et al. (2012) to solve the order batching and routing problem jointly. To the best of our knowledge, this paper is the only one that defines the order picker's capacity constraint as a function of item weight. Another capacity constraint that is different from item count was used by Ho and Tseng (2006) and Ho et al. (2008). They studied several seed-order selection rules and added accompanying-order selection rules to integrate more orders in a batch. The capacity of a batch was limited to 80 cubic units, with the volume of an item ranging from 1 to 3 cubic units.

2.2 Simulated Annealing Approaches Applied to Warehouse Management

Simulated annealing (SA) is a meta heuristic first proposed by Kirkpatrick et al. (1983), and its aim is to solve combinatorial problems by approximating a global optimum at a reasonable computation time. It has been successfully applied to a variety of optimization problems, in particular in production and operations management (see, for example, Eglese 1990;Sarker and Yao 2003; Loukil et al. 2007).In the context of order picking, SA has only infrequently been used so far. Ho and Chien (2006) employed SA to improve a nearest-neighbor heuristic to derive the sequence of the warehouse zonesan order picker should visit. Ho and Tseng (2006) argued that SA may improve order picking routes and applied it to the largest-gap policy. Muppani and Adil (2008) studied the storage assignment problem and developed a SA algorithm to derive efficient storage classes by considering product combinations, storage-space cost and order-picking cost. Chan and Kumar (2009) studied warehouse scheduling with the objective of minimizing lead time and tardiness, and proposed a hybrid tabu search-SA algorithm to solve the problem. Finally, Hong et al. (2012) investing a teda warehouse with several order pickers and used SA to reduce picker blocking and thus total retrieval time.

The literature review shows that item weight has only very infrequently been considered in formulating capacity constraints in order picking models. In addition, SA has successfully been applied in the planning of order picking operations, however, not fully to the order batching problem. In light of these findings, the aim of this paper is to improve the joint optimization of order batching and picker routing by combining commonly used heuristics with a SA approach and by considering an item weight capacity constraint. The aim of this paper is to develop an efficient heuristic that can easily be implemented in practice.

3. Problem Description

This paper studies order picking in a standard picker-to-parts warehouse with parallel aisles, as illustrated in Figure 1.

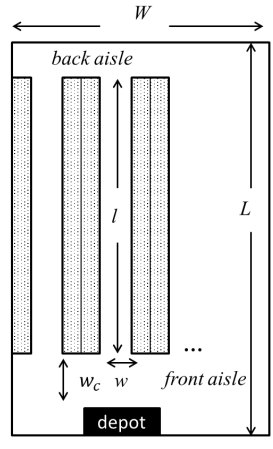

Figure 1 *Warehouse Layout under Study*

As in Figure 1, we assume that the warehouse consists of several picking aisles, front and back access aisles and a depot in the middle of the front aisle. Every pick tour starts and ends at the depot. Cross aisles (front and back) exist only at the respective ends of the picking aisles, and there are no cross aisles in-between. L and W denote the length and width of the warehouse, respectively, and l is the length of the racks, w the width of the gap between two racks, and w_c the width of the front and back aisles. We assume that order picking is performed manually in this warehouse, and that in each storage location, one type of item is stored. Items are assumed to be available on the shelves without limitation. There plenishment of shelves is therefore not considered. The order picking process works as follows: first, the order picker receives an order (on a pick list) at the depot which lists the items to be retrieved. He then starts walking through the aisles and retrieves items from the shelves until his carrying capacity does not permit picking up the item next in sequence anymore. He then returns to the depot to drop-off the items. The process of leaving the depot, picking items and returning to the depot is referred to as 'tour' in the following. After the first tour has been completed, the picker starts the next tour (if necessary) until the customer order is completed. We note that it is possible that the order picker uses a trolley to support the picking process, which would increase the maximum weight he is able to carry in a single tour. We assume that the warehouse is a narrow aisle warehouse, i.e. that the gap that separates two adjacent aisles can be neglected in calculating the travel distance (see also Caron et al. 1998). Moreover, we assume that the horizontal travel distance within aisles is negligible, i.e. that the travel distance does not depend on the height of the shelves or the position of the items on the shelves (Hwang et al. 2004). The storage assignment strategy used in this paper is based on the demand frequency of the items, as this strategy is widely used in practice and has been shown to lead to good results (e.g., Hausmann et al. 1976, Le- Duc and de Koster

2005, Glock and Grosse 2012). Items are thus assigned to storage locations according to their demand frequencies, such that items with high demand are stored in locations near the depot.

4. Model Development

4.1 Definitions

The following assumptions are made:

- Initial tours are constructed with the help of the savings, the return, the midpoint and the largest-gap policies.
- The capacity of the order picker is defined as the maximum (accumulated) item weight the picker is able to carry, C. The picker is not allowed to exceed this capacity in a tour.
- The assignment of items to storage locations is not restricted, e.g. by item type or item weight. Thus, each item can be stored in each storage location.
- The travel speed of the order picker is constant, and a single picker is responsible for picking all orders.
- The travel time is linearly proportional to and an increasing function of the travel distance (de Koster et al. 2007).
- The objective is to minimize the total travel time which is required for picking all orders.
- The rectilinear metric is used for calculating distances in the warehouse.

The following terminology is used throughout the paper:

α	tail index of the Pareto cumulative distribution function
$a_b = (a_{b1}, ..., a_{bn})$	batch vector, where a_{bj} = 1 if order j is included in batch b, and where a_{bj} =0 if order j is not included in batch b
C	capacity of the order picker or pick device in kilogram
c_j	capacity required for picking order j ($j \in J$) in kilogram
$d(i1,i2)$	Rectilinear distance between the storage locations of items $i1$ and $i2$
d_b	total travel distance of the pick tour
D_i	demand coefficient of item i
$F(x)$	Pareto cumulative distribution function
f	transition operator in the SA refinement strategy
g_i	weight of item i
h	transition operator in the SA refinement strategy
i	item i(stock keeping unit)
j	order j
I	set of all feasible batches
$J = (1,...n)$	set of customer orders
k	control parameter in the SA refinement strategy with $k > 0$
K	number of items in an order
L	length of the warehouse
L^S	list that contains the savings values
λ	control parameter in the SA refinement strategy with $\lambda > 0$
l	length of the racks
ΔO	cost difference between two states in the SA refinement strategy

o	depot
ρ	order partition (savings division policy)
s_{i1i2}	distance that is saved when traveling from item $i1$ to item $i2$ instead of traveling from item $i1$ to the depo to and to item $i2$
S	search space in the SA refinement strategy
s	possible solution in the SA refinement strategy with $s \in S$
t_τ	tourτ
T	set of tours
U	uniform random variable in the Pareto cumulative distribution function
W	width of the warehouse
w	gap between two racks
w_c	width of the front and back aisles
x_m	scale index in the Pareto cumulative distribution function
x_b	binary decision variable with ($b \in I$) and ($x_b = 1$) if batch b is selected and ($x_b = 0$) if batch b is not selected

To facilitate the development of the model, the problem under study is divided into five sub-problems, namely: (a) the division policies, (b) the refinement strategy,(c) the definition of item attributes, d) the storage assignment strategy, and e) the routing policies. These sub-problems are discussed separately in the following, and a summary is given in Section 4.7. An extensive numerical study is presented in Section 5.

4.2 The Division Policies

For a given set of orders that need to be picked, the division policies split the orders into batches in such a way that the total weight of all items in a batch does not exceed the capacity of the order picker. Initial order batches are formed by applying four different heuristics, namely a)Clarke and Wright's (1964) savings algorithm, which has frequently been used for this purpose before (e.g. Elsayed and Unal 1989; de Koster et al. 1999), and b) the return, c) the midpoint, and d) the largest-gap heuristic. The policies b), c) and d) have frequently been studied in the order picking literature and are often used in practice for picker routing due to their simplicity (e.g. Goetschalckx and Ratliff 1988;Petersen and Schmenner 1999,Hwang et al. 2004;Petersen and Aase 2004; Theys et al. 2010). As in our experience, only few companies use sophisticated batching algorithms in practice, we decided to computeinitial order batches using the clustering principle of the routing policies discussed in Section 2.1 (for other options to crease initial batches, the reader is referred to the literature review in Section2.1). The division policies return an order partition ρ and are defined as follows:

- The parallel savings algorithm(Clarke and Wright 1964)
  1. For each pair of items $i1 \neq i2$, compute the distance $s_{i1i2} = d(i1, o) - d(i1, i2) + d(o, i2)$ that is saved when the picker travels $i1 \rightarrow i2$ instead of $i1 \rightarrow o \rightarrow i2$.
  2. For $i1, i2 \in [1, K]$, sort s_{i1i2} in descending order in a list L.
  3. Let $\rho = \bigcup_{i=1}^{K}[i]$.
  4. Traverse the list. For each $s_{i1i2} \in L$:

a) Search two tours t_a, $t_b \in \rho$ such that *i1* is the last item of t_a and *i2* is the first item of t_b (or vice versa). If found, then:

b) If t_a and t_b can be joined (that is, if the sum of their total item weights does not exceed the order picker capacity), then:

c) Remove t_a and t_b from ρ.

d) dinsert the concatenation $t_a|t_b$ into ρ (in the vice versa case, $t_b|t_a$).

5. Return ρ.

- The return division policy
  1. Sort, from left to right, all aisles a_n where at least one item has to be picked.
  2. For each a_n, when moving from a_{n-1} to a_n, use the front aisle:
     a) Travel to each item from the front side of the aisle to its back side.
     b) If *C* is reached or all picks have been performed, go to the depot and drop-off all items there.
     c) If there are still items to be picked, return to the next open pick location in the sequence and continue with a).

- The midpoint division policy
  1. Sort, from right to left, all aisles a_n that are located to the left of the depot, and where at least one item under the aisle midpoint (as seen from the depot) has to be picked.
  2. Sort, from left to right, all aisles b_n where at least one item above the warehouse midpoint (as seen from the depot) has to be picked.
  3. Sort, from right to left, all aisles c_n that are located to the right of the depot, and where at least one item under the aisle midpoint has to be picked.
  4. For each a_i (when moving from a_{n-1} to a_n, use the front aisle):

  a) Travel to each item (below the midpoint) from the front side of the aisle to its back side.

  5. For each b_i (when moving from b_{n-1} to b_n, use the back aisle):
     a) Travel to each item (above the midpoint) from the back side of the aisle to its front side.
  6. For each c_n (when moving from c_{n-1} to c_n, use the front aisle):
     a) Travel to each item (below the midpoint) from the front side of the aisle to its back side.
     b) See return policy, Step 2b).

- The largest-gap division policy
  1. For each aisle a_n, let its items be assigned from front to back: $a_n^1, \ldots, a_n^m$. Let a_n^0 and a_n^{m+1} be its intersections with the front and back aisles, respectively. Then, we define its largest gap as the vertical space between the picks a_n^M and a_n^{M+1},

where $M \in [0, \ldots, m]$ is such that it maximizes $d(a_n^M, a_n^{M+1})$.

2. Perform the midpoint policy, substituting 'largest gap' for 'midpoint'.

4.3 The Refinement Strategy

Once orders are split into batches, we use a SA algorithm to search for re-allocations of items between tours that lead to a reduction in the total travel time. The SA algorithm aims at approximating a global optimum in a set *S* of solutions or states (called the *search space*). Each state has an associated numeric value (that here we wish to minimize), representing its cost. Starting from an initial solution, different states are visited iteratively through transition steps in order to progressively decrease the attained cost. SA follows a probabilistic methodology where steps that increase the cost of a solution are accepted with a certain probability. By allowing the algorithm to abandon local minima, this approach improves long-term prospects and raises the chances of eventually reaching a global minimum. The reader is referred to Eglese (1990)for a detailed description of SA.

- **Search Space**. We model our candidate solutions as all possible order batches, i.e. ways to split all orders into a set $(T_i)_i$ of valid tours. The initial solution is the configuration that resulted from the division policy under study, and the SA aims on improving this solution (see Section 4.2).
- **Cost function**. It is a map $h: S \rightarrow R$ defined on the entire search space; in our case, it represents the total travel time $\sum_i t(T_i)$ the picker needs for retrieving all items contained in the order, using the tours defined in the batching step and, for each tour, by considering the given routing policy.
- **Transition Operators**. An operator f is a method to generate possible transitions. Given an input $s \in S$, it produces a set of output states $f(s) \subset S$ that are then used as successor candidates for the next iteration. For our problem, we defined this set as all states that can be produced by taking one item in a tour t_i of the partition *s* and moving it to a different tour t_j. This operator satisfies the prerequisite that all states in the search space are accessible, via one or more transitions, from any initial solution; otherwise, it might not be possible to reach the global optimum.
- **Transition Acceptance**. If *s* is the current state (at the *n*th iteration) and $f(s)$ is the set of its successor candidates, the transition to the next state is evaluated as follows. First, an $s' \in f(s)$ is chosen randomly. If $h(s') \leq h(s)$, then the transition to *s'* implies an improvement and is accepted immediately. Otherwise, let $\Delta C = h(s') - h(s) > 0$ denote the difference of cost between the states. Then the transition is accepted with probability $e^{\frac{-\Delta C}{F(n)}} \in (0, 1)$, where $F(n) = ke^{-\lambda n} > 0$,$k>0$ and $\lambda>0$ are parameters that can be adjusted to control the behavior of the SA algorithm, depending on the specific kind and size of the problem it is applied to (Kirkpatrick et al. 1983). If the transition is rejected, the next iteration takes place at *s* again; the algorithm stops after a predefined number *N* of iterations. Intuitively, this strategy is prone to move to worse solutions at the beginning, thus avoiding poor local minima. In the long term, by temporarily

accepting worse solutions, it is assumed that the solution converges to the global minimum; therefore, a smaller tolerance to higher-cost transitions is preferred as the algorithm approaches termination.

4.4 Defining Item Attributes

To conduct a numerical study in the next section, industrial data and item attributes were derived as follows:

The item weights are automatically generated and follow a Pareto probability distribution. We limit the resulting values by the order picker capacity *C* to ensure that no single item exceeds the maximum weight, which would result in an infeasible problem. The Pareto cumulative distribution function (*cdf*) is given by $F(x) = 1 - (\frac{x_m}{x})^{\propto} \forall x \geq x_m$. We chose $\propto = 1$ (called *tail index*) and $x_m = 1$ (the *scale index*), such that $F(x) = 1 - \frac{1}{x} \in [0,1] \text{for} x \geq x_m = 1$; thus, we obtained each weight (expressed in kilogram) as $g_i = \left(F^{-1}(u), c\right) = min(\frac{1}{1-u}, c)$, where $u \in [0,1]$is a uniformly distributed random variable.

The item demand coefficients D_i (i.e., the probability that a requested item in a given order is item *i*) were obtained likewise employing Pareto pick frequency, with the same parameters α and x_m(Dekker et al. 2004).

4.5 Storage Assignment Strategy

For assigning items to storage locations, we use a modified class-based storage method. Items are assigned to storage location based on their demand frequency as follows:

1. Sort all items i, $i = 1, ..., I$, in descending order according to the demand coefficient D_i.
2. Sort all available storage locations in ascending order according to their distance to the depot *d(o)*.
3. Assign the item with the highest D_i that has not yet been assigned to the storage location with the lowest *d(o)* that is still empty. Mark this item as assigned on the list generated in Step 1, and mark the corresponding storage location as filled on the list generated in Step 2.
4. Repeat Step 3 until all items are assigned to storage locations.

4.6 Routing Policies

For routing order pickers through the warehouse, the heuristics presented in Section 4.2 that were also used for constructing initial batches were used, i.e. the savings, the return, the midpoint, and the largest gap policy.

4.7 Summary

The preceding sections outlined the problem studied and the different steps of the solution procedure used in this paper. We solve the problem sequentially by first splitting orders into batches that can be picked in a single tour and that do not exceed the capacity of the order picker. Initial batches are generated using different division methods that we adopt from the literature on order picker routing. We then improve the batches by using SA to reduce the total travel time. To evaluate our solution procedure in a numerical study, we define item attributes, the storage assignment method and the routing policies.

The objective function (1) and the constraints (2-4) of the problem are formulated as follows (see also Gademann and van de Velde 2005;Henn et al. 2010):

- Travel distance (objective function to minimize): $\sum_{b \in I} d_b * x_b$ (1)
- Each order is included in exactly one batch: $s.t. \sum_{b \in I} a_{bj} * x_b = 1, \forall j \in J;$ (2)
- The batch decision variables are binary: $x_b \in \{0, 1\}, \forall b \in I;$ (3)
- No batch is allowed to exceed the carrying capacity:

$$\sum_{j \in J} c_j * a_{bj} \leq C, \forall b \in I \quad (4)$$

4. Numerical Study

To illustrate the behavior of our model and to assess the efficiency of the division policies in combination with the SA approach, we conducted an extensive numerical study. The layout of the warehouse consists of 10 picking aisles with 1500 pick locations in total. L was set to 28 meters and W was set to 30 meters, with w= 3 meters, w_c = 2 meters and l = 25 meters (see Figure 1). To generate our data set, we assumed that the order picker uses a trolley, and that he has a carrying capacity of C = 100 kg (cf. Kulak et al. 2012). In the simulation study, the order size varies from 10 to 60 items, in steps of size 2. Thus, 10+12+...+58+60 = 910 items were picked for each of the division policies. For each order size, we generated 25 pick lists, leading to a total of 650 pick lists. The total travel time in seconds was calculated for 17 different division policies, i.e. the division policies savings, return, midpoint and largest gap in combination with the four possible routing policies and, as a benchmark, one randomly generated first-come-first-served (fcfs) batching policy. The SA refinement was then applied to each policy, thus leading to a total of 910*25*34=773,500 items that had to be picked. The solution algorithm was programmed with the AIMA-Java software package, and it was run on a ThinkPad laptop with core i5 processor. The necessary data was generated as described in Section 4. The maximum number of iterations for the SA algorithm was set to 5000, and further, after a fine-tuning process had taken place, k was set to 10 and λ was set to 0.05. For each pick list, we calculated the total travel time in seconds and averaged the results. In total, 22,100 experiments were computed.

Table 1 illustrates the average travel time in seconds with order size 50 for the division policies and routing policies savings, return, midpoint, largest-gap, both without SA and with SA refinement, as well as the improvement (reduction in travel time in percent Δ *%)* in case the SA refinement strategy was used. As can be seen, the SA refinement strategy improved all division heuristics on average. Further, it can be seen that the midpoint division policy in combination with the return routing policy led to the highest average travel time for the case where the order batches were generated without SA. Otherwise, this combination achieved the highest improvement in travel time of 14.51% applying the SA refinement. The savings routing policy outperformed all other routing policies regardless of the division policy it was combined with. However, even in case the savings policy was used for division and routing, an improvement could be obtained with the help of SA, although the improvement was small (0.825%). As to possible reductions in total travel time, our numerical study illustrates that the proposed SA approach reduced the

average time the order picker needs to complete an order between 0.825% and 14.51%, as compared to order batching without SA. In the case where the same policy was used for division and routing, the SA approach led to the highest improvement if applied to the midpoint policy(5.229%). Without SA, the average travel time with a random batching and routing policy (fcfs) was 740.92 seconds; applying SA led to an improvement of 44.6% to 410.44 seconds in this case.

Table 1 *Average Travel Time in Seconds for an Order Size of 50*

Division / Routing	**Savings**			**Return**			**Midpoint**			**Largest-Gap**		
Refinement	None	SA	Δ %	None	SA	Δ %	None	SA	Δ %	None	SA	Δ %
savings	239.90	237.92	0.825	308.96	301.80	2.317	283.68	280.80	1.015	274.48	271.42	1.115
return	262.42	245.36	6.501	313.38	301.10	3.919	308.06	288.18	6.453	300.40	279.54	6.944
midpoint	288.20	256.20	11.103	368.02	314.62	14.51	302.56	286.74	5.229	300.82	276.12	8.221
largest-gap	277.08	252.52	8.864	351.08	311.58	11.25	311.44	285.98	8.175	290.60	275.50	5.196

Figures 2(a) to 2(e) plot the improvements in travel time in %against the order size for each division policy and the proposed routing policies. As can be seen, the SA algorithm achieved higher improvements in travel time with an increasing order size in most cases. The marginal improvement, however, was reduced as the order size increased. Table 1 further shows that the division policy created initial batches that were superior to those generated by the random (fcfs) policy. Moreover, SA reduced travel time by almost 45%, as compared to random batching, and by almost 14% as compared to the other division policies. The highest average improvement was achieved by using SA to refine the midpoint division and the return routing policy. One reason for this is that the midpoint and return heuristics led to poorer initial solutions on average with more opportunities for improvement, as compared to the savings algorithm, which offers less room for improvement.

In summary, the SA refinement seems to be a tool that can comfortably and flexibly be used in practice. The computation times for the SA algorithm confirm the practical applicability, as the results for all heuristics and all data sets (including the comparison of percentage improvement by applying SA) could be computed in less than one minute for an order size of 20 items and in 5 minutes for an order size of 40 items. Simulating all results with an order size of 60 could be computed in less than 20 minutes. The results indicate that the improvements that can be achieved by using SA to refine order batches and pick routes may lead to a significant increase in the performance of the warehouse.

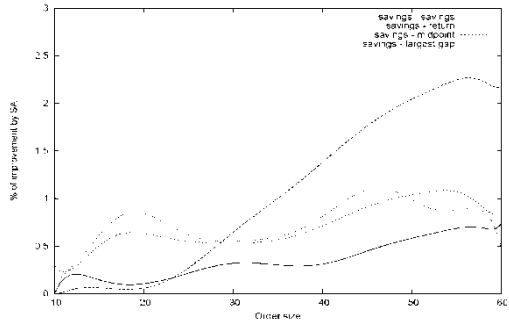

Figure 2(a) *Order Size against Improvements in Travel Time for Savings Division*

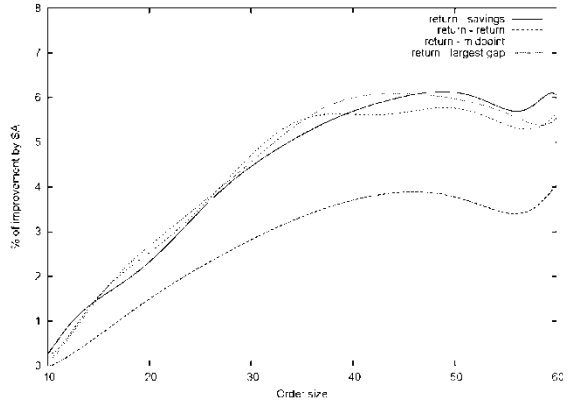

Figure 2(b) *Order Size against Improvements in Travel Time for Return Division*

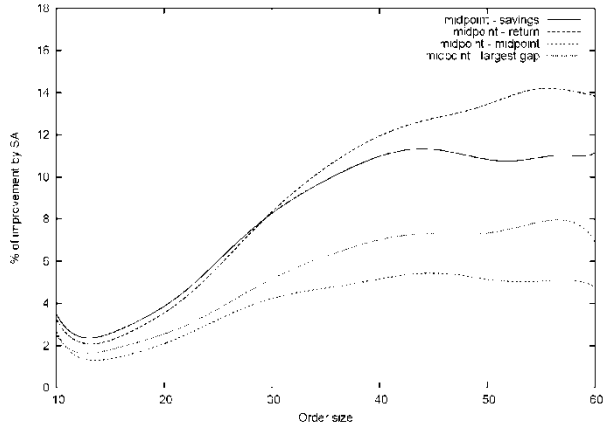

Figure 2(c) *Order Size against Improvements in Travel Time for Midpoint Division*

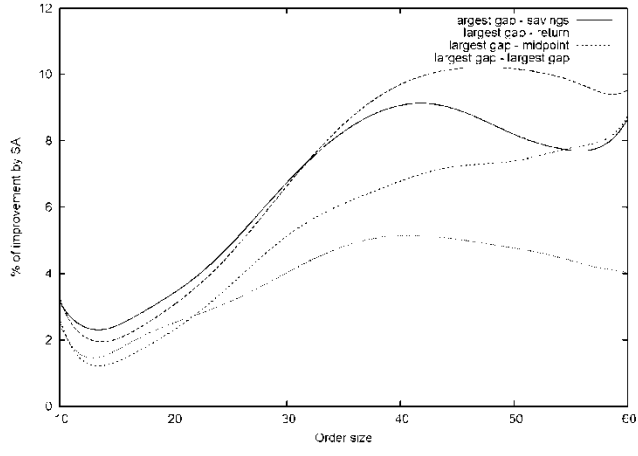

Figure 2(d) *Order Size against Improvements in Travel Time for Largest-Gap Division*

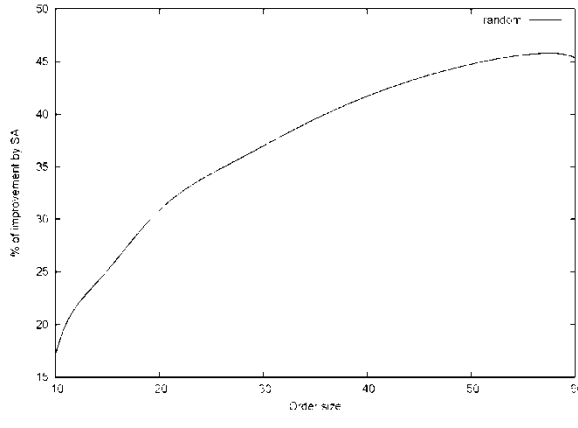

Figure 2(e) *Order Size against Improvements in Travel Time for Random Division*

5. Conclusion

This paper studied an order batching problem in a manual order picking system where the item weight constrains the number of items an order picker is able to carry. Order batches were generated with simple heuristics and improved with a simulated annealing (SA) approach. Numerical experiments showed that the heuristics for order batching and routing studied in this paper can be complemented well by a SA algorithm. The results indicate that the SA refinement strategy can help to improve popular division heuristics, as the average travel time could be reduced by applying SA for all heuristics. With the help of our SA algorithm, managers are able to reduce the average time an order picker needs for completing customer orders. The intention of the paper was to develop an approach that is easy to understand and that can easily be implemented in practice, for example in Java.

This study has limitations. As in most of the existing works, this paper studied order batching and routing sequentially by first deriving order batches and then applying different routing policies to the batches. It is obvious that optimizing both problems

simultaneously will probably lead to better results. Future research could also study the effect of different storage assignment policies on the performance of the SA approach, for example by assigning heavy items to locations close to the depot. In addition, it may be interesting to introduce a sort of a "handling factor" that combines several item characteristics, such as weight, size and volume, or to use different capacity restrictions to arrive at an even more realistic formulation for the order picker capacity. Applying such a model in practice, however, would require a comprehensive data pool that contains item characteristics in addition to demand data, which is often not available. Another promising option to increase the efficiency of the warehouse would be to allow the order picker to pick items that belong to different orders, although this would make sorting operations necessary. Sorting would have a negative effect on the total completion time of an order, but it could be offset by savings in total travel time. In addition, the SA approach developed in this paper could be improved with regard to computation time. Finally, it would be interesting to compare the SA approach to other meta heuristics to gain insights into which meta heuristic is better for solving the order batching problem. We leave these and other extensions for future research.

Acknowledgments: The authors thank the reviewers and the editor for their valuable comments on an earlier version of this paper. They further wish to acknowledge the developers of the AIMA-Java software package, which was of great help in conducting the simulation study.

6. References

1. Bindi, F., Manzini, R., Pareschi, A., and Regattieri, A. (2009),"Similarity-based storage allocation rules in an order picking system: An application to the food service industry", *International Journal of Logistics Research and Applications,* 12(3), pp. 233-247.
2. Bottani, E., Cecconi, M., Vignali, G., and Montanari, R.(2012),"Optimisation of storage allocation in order picking operations through a genetic algorithm", *International Journal of Logistics Research and Applications: A Leading Journal of Supply Chain Management,* 15(2), pp. 127-146.
3. Bozer, Y.A. and Kile, J.W.(2008),"Order batching in walk-and-pick order picking systems", *International Journal of Production Research,* 46(7), pp. 1887-1909.
4. Caron, F., Marchet, G., and Perego, A.(1998). "Routing policies and coi-based storage policies in picker-to-part systems", *International Journal of Production Research,* 36(3), pp. 713-732.
5. Chan, F.T.S. and Kumar, V.(2009),"Hybrid tssa algorithm-based approach to solve warehouse-scheduling problems", *International Journal of Production Research,* 47(4), pp. 919-940.
6. Chen, C.-M. and Wu, H.-P.(2005),"An association-based clustering approach to order batching considering customer demand patterns", *Omega,* 33(4), pp. 333-343.
7. Clarke, G. and Wright, J.W.(1964),"Scheduling of vehicles from a central depot to a number of delivery points", *Operations Research,* 12(4), pp. 568-581.
8. De Koster, R., Van Der Poort, E.S., and Wolters, M.(1999),,,Efficient order batching methods in warehouses", *International Journal of Production Research,* 37(7), pp. 1479-1504.

9. De Koster, R., Le-Duc, T., and Roodbergen, K.J. (2007),"Design and control of warehouse order picking: A literature review", *European Journal of Operational Research,* 182(2), pp. 481-501.
10. Dekker, R., De Koster, R., Roodbergen, K.J., and Van Kalleveen, H.(2004),"Improving order-picking response time at ankor's warehouse", *Interfaces,* 34(4), pp. 303-313.
11. Eglese, R.W.(1990), "Simulated annealing: A tool for operational research", *European Journal of Operational Research* 46(3), pp. 271-281.
12. Elsayed, E.A. and Unal, O.I.(1989),"Order batching algorithms and travel-time estimation for automated storage/retrieval systems", *International Journal of Production Research,* 27(7), pp. 1097-1114.
13. Frazelle, E.A.(2002),World-class warehousing and material handling, New York: McGraw-Hill.
14. Gademann, N., Van Den Berg, J.P., and Van Der Hoff, H.H.(2001),"An order batching algorithm for wave picking in a parallel-aisle warehouse", *IIE Transactions,* 33(5), pp. 385-398.
15. Gademann, N. and Van De Velde, S.(2005),"Order batching to minimize total travel time in a parallel-aisle warehouse", *IIE Transactions,* 37(1), pp. 63-75.
16. Gibson, D.R. and Sharp, G.P.(1992)"Order batching procedures", *European Journal of Operational Research,* 58(1), pp. 57-67.
17. Glock, C.H. and Grosse, E.H.(2012),"Storage policies and order picking strategies in u-shaped order-picking systems with a movable base", *International Journal of Production Research,* 50(16), pp. 4344-4357.
18. Glock, C.H., Grosse, E.H., and Ries, J. M.(in Press),"The lot sizing problem: A tertiary study", *International Journal of Production Economics*, http://dx.doi.org/10.1016/j.ijpe.2013.12.009.
19. Goetschalckx, M. and Ratliff, H.D.(1988),"Order picking in an aisle", *IIE Transactions,* 20(1), pp. 53-62.
20. Grosse, E.H., Glock, C.H., Jaber, M.Y., and Neumann, W.P.(in Press),"Incorporating human factors in order picking planning models: Framework and research opportunities", *International Journal of Production Research.*DOI: 10.1080/00207543.2014.919424.
21. Gu, J., Goetschalckx, M., and Mcginnis, L.F.(2007),"Research on warehouse operation: A comprehensive review", *European Journal of Operational Research,* 177(1), pp. 1-21.
22. Gu, J., Goetschalckx, M., and Mcginnis, L.F.(2010),"Solving the forward-reserve allocation problem in warehouse order picking systems", Journal of the Operational Research Society, 61(6), pp. 1013-1021.
23. Hausman, W.H., Schwarz, L.B., and Graves, S.C.(1976),"Optimal storage assignment in automatic warehousing systems", *Management Science,* 22(6), pp. 629-638.
24. Henn, S., Koch, S., Doerner, K.F., Strauss, C., and Wäscher, G.(2010),"Metaheuristics for the order batching problem in manual order picking systems", *BuR Business Research Journal,* 3(1), pp. 82-105.
25. Ho, Y.-C. and Chien, S.P.(2006),"A comparison of two zone-visitation sequencing strategies in a distribution centre",*Computers & Industrial Engineering*, 50(4), pp. 426-439.

26. Ho, Y.-C. and Tseng, Y.-Y. (2006),"A study on order-batching methods of order-picking in a distribution centre with two cross-aisles", *International Journal of Production Research,* 44(17), pp. 3391-3417.
27. Ho, Y.-C., Su, T.-S., and Shi, Z.-B.(2008),"Order-batching methods for an order-picking warehouse with two cross aisles", *Computers & Industrial Engineering,* 55(2), pp. 321-347.
28. Hong, S., Johnson, A., L., and Peters, B.A.(2012),"Batch picking in narrow-aisle order picking systems with consideration for picker blocking", *European Journal of Operational Research,* 221(3), pp. 557-570.
29. Hsieh, L.-F. and Huang, Y.C.(2011),"New batch construction heuristics to optimise the performance of order picking systems", *International Journal of Production Economics* 131(2), pp. 618-630.
30. Hsu, C.-M., Chen, K.-Y., and Chen, M.-C.(2005),"Batching orders in warehouses by minimizing travel distance with genetic algorithms", *Computers in Industry,* 56(2), pp. 169-178.
31. Hwang, H., Oh, Y.H., and Lee, Y.K. (2004),"An evaluation of routing policies for order-picking operations in a low-level picker to part system", *International Journal of Production Research,* 42(18), 3873-3889.
32. Hwang, H. and Kim, D.G.(2005), "Order-batching heuristics based on cluster analysis in a low-level picker-to-part warehousing system", *International Journal of Production Research,* 43(17), pp. 3657-3670.
33. Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P.(1983),"Optimization by simulated annealing", *Science,* 220(4598), pp. 671-680.
34. Kulak, O., Sahin, Y., and Taner, M.E.(2012), "Joint order batching and picker routing in single and multiple-cross-aisle warehouses using cluster-based tabu search algorithms", *Flexible Services and Manufacturing Journal,* 24(1), pp. 52-80.
35. Le-Duc, T. and De Koster, R.(2005),"Travel distance estimation and storage zone optimization in a 2-block class-based storage strategy warehouse", *International Journal of Production Research,* 43(17), pp. 3561-3581.
36. Loukil, T., Teghem, J., and Fortemps, P. (2007),"A multi-objective production scheduling case study solved by simulated annealing", *European Journal of Operational Research*, 179(3), pp. 709-722.
37. Malmborg, C.J. (1995),"Optimization of cubic-per-order index layouts with zoning constraints", *International Journal of Production Research,* 33(2), pp. 465-482.
38. Muppani, V.R. and Adil, G.K.(2008),"Efficient formation of storage classes for warehouse storage location assignment: A simulated annealing approach", *Omega,* 36(4), pp. 609-618.
39. Parikh, P.J. and Meller, R.D.(2008),"Selecting between batch and zone order picking strategies in a distribution center", *Transportation Research Part E: Logistics and Transportation Review,* 44(5), pp. 696-719.
40. Petersen, C.G. and Schmenner, R.W. (1999),"An evaluation of routing and volume-based storage policies in an order picking operation", *Decision Science,* 30(2), pp. 481-501.
41. Petersen, C.G.(2002)"Considerations in order picking zone configuration", *International Journal of Operations & Production Management,* 22(7), pp. 793-805.
42. Petersen, C.G. and Aase, G. (2004),"A comparison of picking, storage, and routing

policies in manual order picking", *International Journal of Production Economics,* 92(1), pp. 11-19.
43. Petersen, C.G., Aase, G., and Heiser, D.R.(2004),"Improving order-picking performance through the implementation of class-based storage", *International Journal of Physical Distribution & Logistics Management,* 34(7), pp. 534-544.
44. Rattliff, H.D. and Rosenthal, A.S.(1983),"Order picking in a rectangular warehouse: A solvable case of the traveling salesman problem", *Operations Research,* 31(3), pp. 507-521.
45. Roodbergen, K.J. and De Koster, R.(2001),"Routing methods for warehouses with multiple cross aisles" *International Journal of Production Research,* 29(9), pp. 1865-1883.
46. Roodbergen, K.J. and Vis, I.F.A.(2006),"A model for warehouse layout", *IIE Transactions,* 38(10), pp. 799-811.
47. Roodbergen, K.J., Sharp, G.P., and Vis, I.F.A.(2008),"Designing the layout structure of manual order picking areas in warehouses", *IIE Transactions,* 40(11), pp. 1032-1045.
48. Sarker, R. and Yao, X. (2003),"Simulated annealing for solving a manufacturing batch-sizing problem", *International Journal of Operations and Quantitative Management*, 9(1), pp. 65-80.
49. Savino, M.M., Meoli, E., Luo, M. and Wong, M.M.(2010),"Dynamic batch scheduling in a continuous cycle-constrained production system", *International Journal of Services Operations and Informatics*, 5(4), pp. 313-329.
50. Theys, C., Braysy, O., Dullaert, W., and Raa, B.(2010),"Using a tsp heuristic for routing order pickers in warehouses", *European Journal of Operational Research,* 200(3), pp. 755-763.
51. Tompkins, J.A., White, Y.A., Bozer, E.H., and Tanchoco, J.M.A. (2010),Facilities planning, 4th ed. Hoboken, N.J.: John Wiley & Sons.
52. Van Den Berg, J.P.(1999),"A literature survey on planning and control of warehousing systems", *IIE Transactions,* 31(8), pp. 751-762.
53. Won, J. and Olafsson, S.(2005),"Joint order batching and order picking in warehouse operations", *International Journal of Production Research,* 43(7), pp. 1427-1442.

About Our Authors

Eric H. Groess studied Business Administration (Industrial Management and Logistics) at the University of Wurzburg, Germany and works now as a research associate at the Department of Industrial Management at the Technische Universität Darmstadt. His research interests include warehouse optimization, order picking systems, production planning and controlling as well as human factors in logistic systems.

Christoph H. Glock is Professor of Industrial Management at the Technische Universität Darmstadt. His research interests include inventory and warehouse management, supply chain coordination, supply chain organization and supplier selection. Prof. Glock has been published in renowned international journals, such as the International Journal of Production Economics, the International Journal of Production Research or Computers & Industrial Engineering.

Rafael Ballester-Ripoll received MSc degrees in mathematics and computer science

from the Technical University of Catalonia (UPC), in the Center for Interdisciplinary Studies (CFIS). In 2011-12, he was a researcher at the DMAG group from the same university. He is since 2012 a PhD student at the VMML group of the University of Zurich (UZH). His research interests include optimization techniques, machine learning and scientific visualization.