Advances in Intelligent Traffic and Transportation Systems
M. Shafik (Ed.)

doi:10.3233/ATDE230020

Mathematical Models for Order Batching and Assignment in a Deep-Frozen Warehouse

Daniele PRETOLANI[a,1], Xiaochen CHOU[a], Dominic LOSKE[b], Matthias KLUMPP[b] and Roberto MONTEMANNI[a]
[a] *Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Reggio Emilia, Italy*
[b] *Department of Business Administration, Georg-August-University of Göttingen, Göttingen, Germany*

Abstract. Picking operations inside a warehouse are the major share of the total costs of retailing operations. Optimization and harmonization of operations is therefore crucial in the economy of a successful business. In this paper we consider the case of a leading German grocery retailer company and we adapt to their case a Mixed Integer Linear Program solving the order batching, assignment and pickers routing problems. Improvements to the model are also discussed, Computational experiments are finally presented, validating the different models and ranking them in terms of their applicability to real scenarios.

Keywords. Warehouse optimization, order batching, order assignment, case studies, mixed integer linear programming

1. Introduction

An optimized warehouse management system (WMS) is nowadays at the basis of a successful business. Intelligent systems utilizing operations research and advanced analytics techniques are needed to enhance efficiency [1]. For example, it is estimated that the overall picking process typically represents 60% of total cost in a warehouse operation [2] and approximately 50% of a picker's time is spent in traveling with/without articles [3].

A diverse set of activities, including batching, picking, cartonizing, and shipping [4] needs to be optimized in a harmonized way. Depending on the characteristics of the business, the different activities have different peculiarities. For example, in e-commerce the way orders are batched together (batching problem) is very central [5], while in other contexts it is less important. How batches are assigned to the pickers (assignment problem), and how the pickers are routed for order picking (routing problem) are again very much dependent on the layout of the warehouse itself [6]. In this paper we consider the optimization of the hub-warehouse of a large German grocery retailer. The customers in such a case are the different supermarket of the chain, and orders are composed of

[1] Corresponding Author, Daniele PRETOLANI, Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Via Amendola 2, 42122 Reggio Emilia, Italy; E-mail: daniele.pretolani@unimore.

large amounts of goods, that will later be temporarily stored locally at supermarket for short times, or directly going to the shelves for customers to buy them [7].

There are only a few studies that simultaneously consider batching, assignment and routing problems. Chen *et al*. [8] minimized tardiness but they cannot solve real-life instances. Scholz *et al*. [9] presented a method that scales up better. Ardjmand *et al*. [10] finally proposed a Mixed Integer Linear Programming model and some heuristics for the special case of wave-picking (connected to e-commerce). The goal of their model is to minimize the overall makespan for processing a set of orders, given a fixed set of pickers.

In this paper we consider the model described in [10] and we specialize it to the business model of the German grocery retailer that inspired our study, and to the typical layout of their warehouses. We will improve the model and present some experimental results based on realistic orders on a real deep-frozen warehouse.

2. Models

In this section we provide the mathematical models for the minimization of the makespan simultaneously considering batching, assignment and routing problems. First we devise a (slightly) revised version of the general model M described in [10]. Then we tailor the general model for application to a real deep-frozen warehouse, obtaining two different models referred to as M1 and M2. Model M1 is based on a graph representation of the warehouse, and includes an explicit encoding of the routes. Model M2 drops the graph representation and adopts a compact encoding of pickers' tours, as explained later.

2.1. Model M

The problem addressed in [10] assumes a one-to-one relation between item types and item locations (bays, houses), i.e., each type of items is contained in exactly one location, and each location contains exactly one item type. There number of item types/locations is N. There are O orders to be partitioned into batches and assigned to R pickers that can receive at most B batches each. Each batch includes at most C_{order} orders, and requires to pick at most C_{unit} item units. The items required by a batch are picked in a single picker's *tour*; all tours start from the same *entry point* and ends in the same *exit point*. In each tour, an item location is visited at most once.

In the model we use the following sets:

- $\mathcal{B} = \{1,2,\dots,B\}$: set of batches for each picker
- $\mathcal{O} = \{1,2,\dots,O\}$: set of orders
- $\mathcal{R} = \{1,2,\dots,R\}$: set of pickers
- $\mathcal{N} = \{1,2,\dots,N,\}$: set of item types/locations
- $\mathcal{L} = \{0,1,2,\dots,N,G = N+1\} = \mathcal{N} \cup \{0,G\}$: set of nodes

The set of nodes $\mathcal{L}$ is used to represent pickers' tours, and includes the set of item locations $\mathcal{N}$, the entry point 0 and the exit point G.

The model parameters are the following:

- C_{unit}: maximum number of units to pick in a batch
- C_{order}: maximum number of orders in a batch
- $d_{i,j}$: travel time from node I to node j
- $q_{o,j}$: number of items of type j in order o
- $I(o) = \{j \in \mathcal{N} : q_{o,j} > 0\}$: set of item types required by order $o \in \mathcal{O}$

- $Q = \sum_{o=1}^{O} |I(o)|$: total number of item types requests over all orders
- M: sufficiently large number

The following decision variables are introduced in the model:

- $x_{b,o}^r$: 1 if order o is assigned to batch b of picker r, zero otherwise
- $y_{b,i,j}^r$: 1 if node i is visited right before node j in batch b of picker r, zero otherwise
- $t_{b,j}^r$: visit time of node j within batch b of picker r; note that $t_{b,0}^r = 0$ and the time taken by the batch is $t_{b,G}^r$
- C_{max}: total makespan

Model M (revised)

$$\min C_{max} \tag{1}$$

$$\sum_{b=1}^{B} \sum_{r=1}^{R} x_{b,o}^r = 1 \qquad o \in \mathcal{O} \tag{2}$$

$$\sum_{o=1}^{O} x_{b,o}^r \leq C_{order} \qquad b \in \mathcal{B}, r \in \mathcal{R} \tag{3}$$

$$\sum_{o=1}^{O} \sum_{i=1}^{N} x_{b,o}^r \, q_{o,i} \leq C_{unit} \qquad b \in \mathcal{B}, r \in \mathcal{R} \tag{4}$$

$$x_{b,o}^r \leq \sum_{i=0, i \neq j}^{N} y_{b,i,j}^r \qquad o \in \mathcal{O},\ b \in \mathcal{B},\ r \in \mathcal{R},\ j \in I(o) \tag{5}$$

$$\sum_{i=0}^{N} y_{b,i,j}^r = \sum_{k=1}^{G} y_{b,j,k}^r \qquad b \in \mathcal{B}, r \in \mathcal{R}, j \in \mathcal{N} \tag{6}$$

$$\sum_{j=1}^{G} y_{b,0,j}^r = 1 \qquad b \in \mathcal{B}, r \in \mathcal{R} \tag{7}$$

$$\sum_{i=0}^{N} y_{b,i,G}^r = 1 \qquad b \in \mathcal{B}, r \in \mathcal{R} \tag{8}$$

$$t_{b,j}^r \geq d_{0,j} \, y_{b,0,j}^r \qquad b \in \mathcal{B}, r \in \mathcal{R}, j \in \mathcal{N} \tag{9}$$

$$t_{b,j}^r \geq t_{b,i}^r + d_{i,j} - M\left(1 - y_{b,i,j}^r\right) \qquad b \in \mathcal{B}, r \in \mathcal{R},\ i, j \in \mathcal{N} \tag{10}$$

$$t_{b,G}^r \geq t_{b,i}^r + d_{i,G} \, y_{b,i,G}^r \qquad b \in \mathcal{B}, r \in \mathcal{R}, i \in \mathcal{N} \tag{11}$$

$$C_{max} \geq \sum_{b=1}^{B} t_{b,G}^r \qquad r \in \mathcal{R} \tag{12}$$

$$y_{b,i,j}^r \in \{0,1\} \qquad b \in \mathcal{B},\ r \in \mathcal{R},\ i, j \in \mathcal{L}$$

$$x_{b,o}^r \in \{0,1\} \qquad o \in \mathcal{O},\ b \in \mathcal{B},\ r \in \mathcal{R}$$

$$t_{b,j}^r \geq 0 \qquad b \in \mathcal{B},\ r \in \mathcal{R},\ i \in \mathcal{L}$$

Constraints (2) require that each order is included in one batch, while (3) and (4) are the capacity constraints for a batch, respectively in terms of number of orders and of units. Constraints (5) enforce that if a batch requires item type j than the corresponding tour must visit location j. Flow-conservation constraints (6)-(8) define the structure of a tour, while constraints (9)-(11) define the visit time for each location within each tour. Note that (10) play the role of *subtour-elimination constraints*, enforcing that a tour (as defined by variables y) is a simple path from the entry to the exit point. If batch b of a picker r is empty (i.e., r is assigned less than B batches) the corresponding tour is represented by setting $y_{b,0,G}^r = 1$, with $t_{b,G}^r = 0$. Finally, (12) define the total makespan as the maximum makespan over all pickers. The number of variables and constraints are $O(B \cdot R \cdot (O + N^2))$ and $O(B \cdot R \cdot (Q + N^2))$, respectively.

2.2. Tailoring Model M for a Real Warehouse

The layout of the particular warehouse addressed in our work is described in Fig. 1. The warehouse consists of seven parallel *aisles*, numbered left to right from one to seven. Each aisle contains 60 houses, 30 located on the right side and 30 on the left side of the aisle. Overall we have $N = 420$ houses, numbered as suggested in Figure 1: aisles are considered left to right, and in each aisle, left houses are progressively numbered bottom-up, and then right houses are numbered top-down. Note that houses are numbered in order of visit, i.e., in a picker's tour house i is visited before every house j>i. Indeed, movements inside the warehouse follow a sharp *routing policy* that (for easiness of explanation) can be summarized as follows: a picker enters an aisle from the bottom left corner, makes a U-shaped tour, and leaves the aisle from the bottom right corner, either proceeding towards an aisle further right or leaving the warehouse. Pickers move from one aisle to another along the *main corridor* placed under the aisles; note that the entry and exit points are located on the left side of the main corridor. Consider the example in Figure 1: a picker's U-shaped tour inside aisle five starts from house 241 and ends in house 300, after that, the picker may move to aisle six or seven, or go to the exit point.

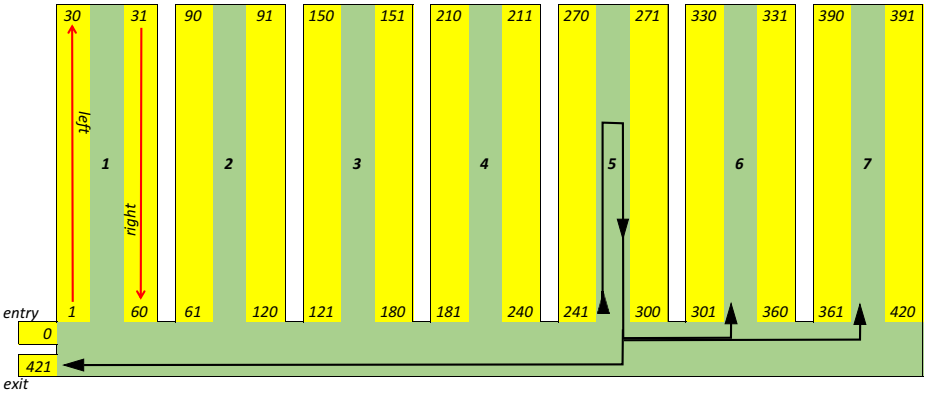

Figure 1. Warehouse layout, item location numbering and picker movements.

To describe the layout of the warehouse, and the movements of pickers inside it, we introduce an acyclic directed graph $W = (\mathcal{L}, \mathcal{M})$. The node set is $\mathcal{L} = \mathcal{N} \cup \{0, G\}$ as defined earlier. The set of arcs $\mathcal{M}$ is partitioned into three subsets $\mathcal{M}^{(1)}$, $\mathcal{M}^{(2)}$ and $\mathcal{M}^{(3)}$, corresponding to different types of movements. With reference to Fig. 1, arcs in $\mathcal{M}^{(1)}$ represent vertical movements along the left side (up) or right side (down) of an aisle; arcs in $\mathcal{M}^{(2)}$ represent U-turns inside aisles, i.e., crossing the aisle left to right; and arcs in $\mathcal{M}^{(3)}$ represent horizontal movements from a visited aisle, or the entry point, to the next aisle to visit, or the exit point. To formally define the three subsets we need to introduce some further notation; let

- $\mathcal{L}^0 = \{j \in \mathcal{L}: (j \bmod 60) = 0\} = \{0, 60, 120, \cdots, 420\}$
- $\mathcal{L}^1 = \{j \in \mathcal{L}: (j \bmod 60) = 1\} = \{1, 61, 121, \cdots, 421\}$
- $\mathcal{N}^1 = \{j \in \mathcal{N}: 1 \leq (j \bmod 60) \leq 30\}$

The set $\mathcal{N}^1$ contains the houses on the left side of the aisles, and given a $j \in \mathcal{N}^1$, the house in front of j on the right side is $U(j) = j + 61 - 2 \cdot (j \bmod 60)$. Then we have $\mathcal{M} = \mathcal{M}^{(1)} \cup \mathcal{M}^{(2)} \cup \mathcal{M}^{(3)}$ where:

- $\mathcal{M}^{(1)} = \{\ (j, j+1) : j \in \mathcal{N},\ (j \bmod 30) \neq 0\ \}$
- $\mathcal{M}^{(2)} = \{\ (j, U(j)) :\ j \in \mathcal{N}^1\ \}$
- $\mathcal{M}^{(3)} = \{\ (j,k) :\ j \in \mathcal{L}^0,\ k \in \mathcal{L}^1,\ k > j\ \}$

It is easy to see that each possible tour of a picker corresponds to a directed path from 0 to G in graph W, in particular, the single arc $(0, G) \in \mathcal{M}^c$ represents a fictitious tour corresponding to an empty batch. This graph representation is exploited in model M1. Moreover, note that a picker's tour is completely described by the set of visited aisles and by the turning points inside them, i.e., where the picker crosses the aisle going from j to $U(j)$. This fact is exploited in model M2 to devise a compact representation of the tours and their durations.

2.3. Model M1

Similar to model M, model M1 represents pickers' tours as directed paths from *0* to G in graph W, by means of the y variables. However, the meaning of y variables is slightly different here, namely, $y^r_{b,i,j}$ is 1 if and only if arc $(i,j) \in \mathcal{M}$ belongs to the path corresponding to batch b of picker r. In other words, $y^r_{b,i,j} = 1$ does not necessarily imply that some items are picked from house j.

Moreover, the representation is significantly simplified. Being W acyclic, we no longer need subtour elimination constraints, thus we can drop the explicit time variables $t^r_{b,j}$ and the constraints (9)-(11). The total travel time for a batch is obtained as the sum of the travel times d on the arcs of the corresponding tour; we replace constraints (12) by constraints (17) accordingly.

In order to represent graph W in a compact way in M1 we define for each node $j \in \mathcal{L}$ the set of outgoing arcs (*forward star*) $FS(j)$ and the set of incoming arcs (*backward star*) $BS(j)$. Constraints (5)-(8) are rewritten accordingly into (13)-(16).

Model M1

$$\min C_{max} \tag{1}$$

s.t. (2)-(4)

$$x^r_{b,o} \leq \sum_{(i,j) \in BS(j)} y^r_{b,i,j} \qquad o \in \mathcal{O}, b \in \mathcal{B}, r \in \mathcal{R}, j \in I(o) \tag{13}$$

$$\sum_{(i,j) \in BS(j)} y^r_{b,i,j} = \sum_{(j,k) \in FS(j)} y^r_{b,j,k} \qquad b \in \mathcal{B}, r \in \mathcal{R}, j \in \mathcal{N} \tag{14}$$

$$\sum_{(0,j) \in FS(0)} y^r_{b,0,j} = 1 \qquad b \in \mathcal{B}, r \in \mathcal{R} \tag{15}$$

$$\sum_{(i,G) \in BS(G)} y^r_{b,i,G} = 1 \qquad b \in \mathcal{B}, r \in \mathcal{R} \tag{16}$$

$$C_{max} \geq \sum_{b=1}^{B} \sum_{(i,j) \in \mathcal{M}} d_{i,j} y^r_{b,i,j} \qquad \forall\, r \in \mathcal{R} \tag{17}$$

$$y^r_{b,i,j} \in \{0,1\} \qquad b \in \mathcal{B}, r \in \mathcal{R},\ (i,j) \in \mathcal{M}$$

$$x^r_{b,o} \in \{0,1\} \qquad b \in \mathcal{B}, r \in \mathcal{R},\ o \in \mathcal{O}$$

Being W a sparse graph, such that $|\mathcal{M}| = O(N)$, the number of variables drops to $O(B \cdot R \cdot (O + N))$, while the number of constraints drops to $O(B \cdot R \cdot (Q + N))$.

2.4. Model M2

In model M2 we drop the explicit representation of the pickers' tours by means of y variables. Instead, we introduce new variables to keep track of the visited aisles and of the *length* of the U-shaped walks inside them: for a picker crossing aisle a from house j to $U(j)$ the length of the walk inside a is $(j \bmod 60)$. More precisely, we assume that the time taken by a tour is obtained as the sum of the following components:

- for each visited aisle there is a fixed time t_e for entering/exiting the aisle, i.e. going from the main corridor to the bottom-left house and back;
- a U-shaped walk of length v requires time $(v-1) \cdot T_D$ for moving up (from bottom-left house $K+1$ to $K+v$) and down (from $U(K+v)$ to bottom-right house); here $T_D < t_e$ is twice the time to move from a house to the next one on the same side;
- given that a is the last (i.e., rightmost) visited aisle, there is a further time T^a_{last} that accounts for the horizontal movements, i.e. the distance between a and the entry/exit points; we have $T^a_{last} = 2d_{0,j}$, where j is the bottom-right house in a.

To write the model we introduce the set of aisles $\mathcal{A} = \{1,2,\cdots,7\}$ and for each order $o \in \mathcal{O}$ and aisle $a \in \mathcal{A}$ we let $D_{o,a}$ be the minimum length of a walk inside aisle a required by order o. We assume $D_{o,a} = 0$ if aisle a does not need to be visited to satisfy order o. Since there are 30 houses on each side of an aisle, we have $0 \le D_{o,a} \le 30$, in particular $D_{o,a} = \max\{0, H_{o,a}\}$ where:

$$H_{o,a} = \max\{(j \bmod 60):\ \lceil j/60 \rceil = a,\ j \in \mathcal{N}^1,\ j \in I(o) \ \vee\ U(j) \in I(o)\}$$

We let $T_A = t_e - T_D > 0$, and we introduce the following sets of new variables:

- $u^r_{b,a}$: 1 if batch b of picker r requires aisle a, zero otherwise
- $w^r_{b,a}$: 1 if a is the last aisle required by batch b of picker r, zero otherwise
- $v^r_{b,a}$: length of the U-shape walk inside aisle a for batch b of picker r, zero if a is not entered

Model M2

$$\min C_{max} \tag{1}$$

s.t. (2)-(4)

$$D_{o,a} x^r_{b,o} \le v^r_{b,a} \qquad b \in \mathcal{B}, r \in \mathcal{R}, o \in \mathcal{O}, a \in \mathcal{A} \tag{18}$$

$$v^r_{b,a} \le 30 \cdot u^r_{b,a} \qquad b \in \mathcal{B},\ r \in \mathcal{R},\ a \in \mathcal{A} \tag{19}$$

$$u^r_{b,a} \le \sum_{e=a}^{7} w^r_{b,e} \cdot \qquad b \in \mathcal{B},\ r \in \mathcal{R},\ a \in \mathcal{A} \tag{20}$$

$$C_{max} \ge \sum_{b=1}^{B} \sum_{a \in A} \left(T_A \cdot u^r_{b,a} + T_D \cdot v^r_{b,a} + T^a_{last} \cdot w^r_{b,a} \right) \qquad r \in \mathcal{R} \tag{21}$$

$$x^r_{b,o} \in \{0,1\} \qquad b \in \mathcal{B}, r \in \mathcal{R}, o \in \mathcal{O}$$

$$u^r_{b,a},\ w^r_{b,a} \in \{0,1\} \qquad b \in \mathcal{B},\ r \in \mathcal{R},\ a \in \mathcal{A}$$

$$v^r_{b,a} \ge 0 \qquad b \in \mathcal{B},\ r \in \mathcal{R},\ a \in \mathcal{A}$$

Constraints (18) set variables v; constraints (19) set $u_{b,a}^r = 1$ if batch b of picker r requires aisle a; constraints (20) set $w_{b,a}^r = 1$ if a is the last aisle required by batch b of picker r: note that in an optimal solution at most one variable $w_{b,a}^r$ will be set to one for each batch b and picker r. Finally, contraints (21) compute the makespan as described above.

Both the number of variables and the number of constraints fall to $O(B \cdot R \cdot O)$. Note that M2 can be generalized to an arbitrary number A of aisles; in this case the number of variables is $O(B \cdot R \cdot (O + A))$ and the number of constraints is $O(B \cdot R \cdot O \cdot A)$.

3. Computational Results

In this section we present computational experiments with simulations of order picking tasks in different scenarios. We compare the performance of the three models described in Section 2 in terms of solvability under different conditions and execution time.

The models are implemented in Python 3.8 and solved with Gurobi[2] 9.5.1. All the experiments were carried out on a computer with 3.5GHz Intel Core i7 processor and 16GB of RAM.

3.1. The Limitations of Model M

First, we test the models on a small dataset of 5 orders that contains 20 items in total, with items all located in at most two aisles. As described in Section 2.2, an aisle for the warehouse studied in this work contains 60 houses on each side. The items are randomly assigned to a given number of locations $n = \{5, 10, 15, 20, 60, 120\}$. The number of pickers is set to 2. The instances are then solved with a time limit of 300 seconds for each instance. When no optimal solution can be found within the time limit, the instance is marked as out-of-time. The results are reported in Table 1.

Table 1. Solving time (s) of Model M, M1 and M2 on a small dataset with different number of locations *n*

n	**5**	**10**	**15**	**20**	**60**	**120**
M	0.371	15.663	out-of-time	out-of-time	out-of-time	out-of-time
M1	0.003	0.003	0.004	0.004	0.004	0.007
M2	0.001	0.001	0.001	0.001	0.001	0.002

From Table 1 we can observe that Model M is extremely sensible to the size of the instance. This limits the further use of it to the problem we study. Meanwhile both M1 and M2 are able to solve the small instances within a very short amount of time. We therefore continue the experiments on the standard layout with 7 aisles (see Section 2.2) for both models and further increase the number of orders to study their solving limits.

3.2. Number of Orders that the Models Can Handle

To measure the quantity of picking tasks, the number of orders is considered intuitively in practice. Since each order contains a different number of items and each item can be

[2] https://www.gurobi.com/

picked multiple times, the total number of units to be picked is set as another indicator. Three different scenarios are considered for experiments: In the first scenario, each order contains only one item, we refer to this set of instances as *Single*. In the second scenario, each order contains a random number of items in the interval [1, 5], we refer to this set of instances as *Middle*. In the third scenario, each order contains a random number of items in [1,10], we refer to this set of instances as *Multiple*. In all scenarios the required quantity of each item is generated with a random number in [1, 10]. The number of pickers is set to 4. In a single batch, a picker can handle at most 3 orders and a number of units given by the maximum possible number of units in an order, namely 10, 50 and 100 for scenarios *Single*, *Middle* and *Multiple*, respectively.

The instances are solved by Model M1 and M2 with a time limit of 300 seconds. We report the number of orders, the type of scenario, the total number of units to be picked, the minimum total Makespan found by the solver and the solving time for each model in Table 2.

Table 2. Performance of Model M1 and M2 on instances with different number of orders *O*

O	**9**	**12**	**15**	**18**	**21**
			Single		
Total units	54	68	87	98	103
Makespan	268	326	364	398	446
Time M1	2.63	33.42	out-of-time	out-of-time	out-of-time
Time M2	0.24	2.21	26.34	47.49	out-of-time
			Middle		
Total units	168	201	268	294	324
Makespan	446	572	684	730	770
Time M1	5.97	24.38	249.58	out-of-time	out-of-time
Time M2	0.15	0.38	1.49	11.97	101.31
			Multiple		
Total units	220	299	395	526	633
Makespan	568	594	722	966	1080
Time M1	0.42	0.71	16.29	out-of-time	out-of-time
Time M2	0.03	0.09	3.74	7.96	30.44

In Table 2 we can observe that M2 performs better than M1 in terms of solving time. The upper limit of the number of orders that can be solved by M1 is around 12 to 15 depending on the scenarios, while the limit for M2 is around 21. We can also observe that the solving time of the models is not dominated by the total number of units but the number of orders and type of scenarios. In general, we can conclude that both models are able to provide the optimal solution for task assignment in applicable daily scenarios in a short amount of time, while model M2 offers a faster computing speed. To further optimize the task assignment, in the next section we will compare and suggest the appropriate number of pickers for a given order.

3.3. Number of Pickers

An example instance with 12 orders generated with the *Middle* scenario is considered in this section. We test the performance of the model with different number of pickers varying from 2 to 7. Each picker can deal with no more than 3 orders and 50 units for one batch. The minimum total Makespan (optimal) and the number of batches of the task assignment solution found for different number of pickers are reported in Table 3. We also compare the performance in terms of computing time for model M1 and M2.

Table 3. Comparison of Model M1 and M2 on an example instance with different number of pickers

	2	3	4	5	6	7
Makespan	1002	714	572	424	388	384
Batches	4	3	2	2	1	1
Time M1	out-of-time	27.22	55.26	12.58	0.36	0.52
Time M2	0.49	0.52	0.75	0.23	0.07	0.09

In Table 3 we can observe that the optimal total Makespan gradually decreases as the number of pickers increases. Each picker only needs to complete one batch when the number of pickers reaches 6. While the labor cost should be comprehensively calculated in practical applications, here we can roughly consider that it is not a reasonable arrangement to set more than 6 pickers for such a given problem size, for the reason that a further increase on the number of pickers will increase only the labor cost, but not reduce the optimal minimum Makespan in an effective manner. In terms of solving speed, M1 is able to solve the instance with a reasonable number of pickers between 3-5 within one minute while M2 is able to give an optimal solution for all settings with less than a second. This shows again that when the quality of the solution is the same, M2 is much faster in terms of computing speed.

From the conclusions of the previous section, both M1 and M2 will reach their respective upper limits as the number of orders grows. Therefore, to further test their potentials, we run experiments with the linear relaxation of the models in the next section.

3.4. Linear Relaxation of Model M1 and M2

In this section, we test the linear relaxation of Model M1 and M2 with large number of orders $O = \{40, 60, 80, 100, 120\}$. The instances are generated with the scenario *Middle* that has a random number of items between [1, 5] in each order. The quantity of each item is generated between [1, 10] as mentioned in Section 3.2.

First, we generate a set of instances with 4 pickers, where each picker can deal with no more than 3 orders and 50 units for one batch. The total number of orders are {40, 60, 80, 100}. This set of instances is referred to as *Large I*. We then scale up the instances to 8 pickers, where each picker can deal with no more than 6 orders and 100 units for one batch. The total number of orders are {60, 80, 100, 120}. This set of instances is referred to as *Large II*. To these two datasets we apply the ILP solver setting a CPU limit of ten seconds, which in many cases suffices to solve the LP relaxation and a few nodes of the enumeration tree. In these cases the solver possibly returns a feasible solution (either found heuristically or during the enumeration) and the best lower bound found, together with the resulting optimality gap. We report the optimum value and CPU time for the LP relaxation, the best feasible solution, best lower bound and gap (%) and the number of batches for both datasets in Table 4 and 5. The instance is marked as out-of-time if the LP relaxation is not solved within the given time limit. For the sake of completeness, in a couple of cases (dataset *Large I*, 60 and 80 orders) we forced the solution of the LP relaxation dropping the CPU time limit.

Table 4. Performance for the linear relaxation of Model M1 and M2 on large dataset *Large I*

			Model M1			
O	**LP opt**	**CPU time**	**Best sol**	**Best bound**	**Gap (%)**	**Batches**
40	326	2.22	1846	438	76.3	5
60	*360.38*[(a)]	out-of-time	-	-	-	7
80	*376.43*[(a)]	out-of-time	-	-	-	9
100	396.08	10	-	397	-	12
			Model M2			

O	LP opt	CPU time	Best sol	Best bound	Gap (%)	Batches
40	229.5	0.41	1694	738	56.5	5
60	233.5	0.94	2898	354	87.8	7
80	236.5	6.58	4472	238	94.7	9
100	236.5	8.36	5420	238	95.6	12

[(a)] *Computed dropping the CPU time limit*

Table 5. Performance of the linear relaxation model of Model M1 and M2 on large dataset *Large II*

			Model M1			
O	**LP opt**	**CPU time**	**Best sol**	**Best bound**	**Gap (%)**	**Batches**
60	116.75	0.82	1460	270	81.5	2
80	118.25	1.38	1604	135	91.6	2
100	118.25	4.01	2498	119	95.2	3
120	119.25	7.63	2482	120	95.2	3
			Model M2			
O	**LP opt**	**CPU time**	**Best sol**	**Best bound**	**Gap (%)**	**Batches**
60	116.75	0.36	890	474	46.7	2
80	118.25	0.63	1178	474	59.8	2
100	118.25	3.5	1786	120	93.3	3
120	119.25	4.99	2026	120	94.1	3

From Table 4 and 5 we can observe that the gap is large for the relaxation models, i.e. the relaxation bound is rather poor, and further work is expected in order to improve the performance of the models on large datasets. Remarkably, the relaxation bound provided by model M1 is never worse than the one of M2, and is better for dataset *Large I*. On the other side, M2 requires less time for solving the linear relaxation, and allows to find much better feasible solutions in short time. These results seem to indicate that both models deserve to be considered for further analysis, even if M2 seems to have higher potential to be adapted to large number of orders, possibly in a matheuristic fashion.

4. Conclusion

We considered the operational optimization of a hub-warehouse of a large German grocery retailer. In particular, we addressed the minimization of the order picking makespan, that simultaneously considers the batching, assignment and routing problems. Starting from a MILP model proposed in the literature, we devised new models specifically tailored for the above warehouse. We showed that the proposed models are clearly faster than the original model for the specific application and can find optimal solutions for relatively small sets of realistic orders.

The above results are quite encouraging since they show that tighter and faster models can be devised for the specific layout and routing policy of the considered warehouse. Clearly, further work is needed to deal with large sets of orders. This include in particular the development of Lagrangean decomposition methods and Lagrangean heuristics, both based on the new formulations presented here. A further direction is given by (meta-)heuristic methods to be designed *ad hoc* for the particular warehouse.

We remark that our models can be easily extended to consider picking times, in addition to travel times; moreover, the specific skills of each picker may be taken into consideration in the definition of the time taken by each operation, see e.g. [11]. Finally, the approach followed in models M1 and M2 may be adapted to warehouses with different layouts, provided that the movements of the pickers follow a specified routing policy, see the examples considered in [12].

References

[1] Binder C, Neureiter C, Lastro G. Towards a Model-Driven Architecture Process for Developing Industry 4.0 Applications. International Journal of Modeling and Optimization 2019; 9(1):1–6.
[2] Kulak O, Sahin Y, Taner ME. Joint order batching and picker routing in single and multiple-cross-aisle warehouses using cluster-based tabu search algorithms. Flex. Serv. Manuf. J. 2012; 24:52–80.
[3] Tompkins JA, White JA, Bozer YA, Tanchoco JMA. Facilities Planning. John Wiley & Sons; 2010.
[4] Ardjmand E, Bajgiran OS, Rahman S, Weckman GR, Young WA. A multi- objective model for order cartonization and fulfillment center assignment in the e- tail/retail industry. Transport. Res. Part E Logist. Transport. Rev. 2018; 115:16–34.
[5] Boysen N, de Koster R, Weidinger F. Warehousing in the e-commerce era: A survey. European Journal of Operational Research 2019; 277(2):396-411.
[6] Batt RJ, Gallino S. Finding a needle in a haystack: The effects of searching and learning on pick-worker performance. Management Science 2019; 65(6):2624-2645.
[7] Boysen N, de Koster R, Füßler D. The forgotten sons: Warehousing systems for brick-and-mortar retail chains. European Journal of Operational Research 2021; 288(2):361-381.
[8] Chen TL, Cheng CY, Chen YY, Chan LK. An efficient hybrid algorithm for integrated order batching, sequencing and routing problem. Int. J. Prod. Econ. 2015; 159:158–167.
[9] Scholz A, Schubert D, Wäscher G. Order picking with multiple pickers and due dates–Simultaneous solution of Order Batching, Batch Assignment and Sequencing, and Picker Routing Problems. Eur. J. Oper. Res. 2017; 263(2):461–478.
[10] Ardjmand E, Shakeri H, Singh M, Sanei Bajgiran O. Minimizing order picking makespan with multiple pickers in a wave picking warehouse. International Journal of Production Economics 2018; 206:169-183.
[11] Matusiak M, de Koster R, Saarinen J. Utilizing individual picker skills to improve order batching in a warehouse. European Journal of Operational Research 2017; 263:888-899.
[12] Pan JCH, Wu MH, Chang WL. A travel time estimation model for a high-level picker-to-part system with class-based storage policies, European Journal of Operational Research 2014; 237:1054-1066.