Management and Production Engineering Review
Volume 15 • Number 1 • March 2024 • pp. 140–160
DOI: 10.24425/mper.2024.149996

Matheuristics for the Order-picking Problem with Sequence-dependant Constraints in a Logistic Center with a One-directional Conveyor Between Buffers

Kateryna Czerniachowska[1], Radosław Wichniarek[2], Krzysztof Żywicki[2]

[1] *Wroclaw University of Economics and Business, Wroclaw, Poland*
[2] *Poznan University of Technology, Poznan, Poland*

Received: 24 October 2023
Accepted: 04 February 2024

Abstract
In the logistics center (warehouse or distribution center), customer orders need to be picked up by the pickers. In this research, we examine the order-picking problem with sequence-dependent constraints with two decision variables (container start time and product quantity) in a distribution center with a one-directional conveyor. The decision-making is based on the developed two variations of two-step matheuristics. At first, the main order-picking problem is divided into two subproblems. Next, each step of each variant of the subproblem is solved using a mathematical programming-based technique. Both matheuristics were better in 85 of 120 test instances compared to the initial model solved by mathematical programming. Pickers matheuristics were better on average at 46.56%, while Buffers matheuristics were better on average at 46.87%. The proposed matheuristics approach allows distributors to schedule orders in the logistics center fast enough and with fewer resources.

Keywords
order picking; mathematical programming; matheuristics; logistics center; distribution center.

Introduction

The manufacturing industry has seen numerous significant changes, and the introduction of new technology is changing how people are involved in the production process (Beauchemin et al., 2022).

One of an industrial company's most crucial tasks in order to maintain its competitiveness in the market is production management (Fuchigami and Rangel, 2018). Effective production schedule management lowers production costs, makes better use of resources, and improves the quality of the service (Da Col and Teppan, 2022).

Based on the availability of resources and customer orders, businesses can modify their production scheduling. A production schedule aims to operate cost-effectively while maintaining a balance between client needs and the resources at hand. The warehouse or distribution centers (DC) will encounter difficulties picking orders and meeting the delivery deadlines if the production schedule is not precise and realistic to the given available resources.

In recent years, mail-order business via the Internet has benefited greatly from large-scale logistics centers. Quick product delivery to clients in the logistics center depends on effective management. There have been numerous studies on how to effectively regulate the logistics center (Iwasaki et al., 2013).

Several multinational corporations constructed distribution centers across Europe at the same time. Typically, these DCs are in charge of delivering items made outside of Europe to clients in Europe as part of business-to-consumer (B2C) e-commerce transactions (Hultkrantz and Lumsden, 2001).

The fulfillment center receives, stores, and organizes inventory until it is requested. Following receipt of an order, the fulfillment center selects, packages, and ships an item to the final recipient.

Online consumer orders are transformed into delivery packages by fulfillment warehouses. The main distinctions between fulfillment warehouses and a regular warehouse are an enormous quantity of tiny storage bins, an explosive storage policy that distributes each

***Corresponding author**: Krzysztof Żywicki – Poznan University of Technology, Faculty of Mechanical Engineering, Piotrowo 3 Street, Poznan, Poland, 61-138, phone: +48 61 665-27-40, e-mail: krzysztof.zywicki@put.poznan.pl*

item's inventory among several bins, and commingled bin storage. The highly volatile storage picking problem, a variant of the conventional order-picking problem, is introduced by Onal et al. (2023). There are several picking possibilities available for every item because they stock them in several locations.

One of the most popular services in e-commerce retail is quick delivery. It entails quick delivery of online-ordered merchandise. In this market, customer orders have deadlines, and adhering to them is essential to providing high-quality services. Order picking takes up the greatest time in the warehouse. It entails aggregating orders into batches, giving order pickers access to those batches, and sequencing the batches given to each order picker in a way that both meets order deadlines and cuts down on picking time. E-commerce warehouses use new logistical techniques to expedite the order-picking processes (Haouassi et al., 2022).

Additionally, if the production plan was not produced quickly enough, it can happen that orders were packed after the scheduled shipment time, and they must wait for the next shipment date and time, so the plan was already out of date. In this article, we outline a method for approaching the issue in the warehouse and DC with matheuristics so that the scheduling outcome is acceptable for both small and large-scale businesses.

This article's scientific contribution then consists of the following:

- Proposing the model for the logistics order-picking problem with sequence-dependent constraints and a one-directional conveyor with two decision variables: container start time and product quantity. This model differs from the research by Czerniachowska et al. (2023 a,b) with a sequence of products that should be added to the container.
- Simplifying the model by excluding not-ordered products from all constraints. This is also a feature that differs this model from Czerniachowska et al. (2023 a,b).
- Developing the two variants of matheuristics to solve the proposed order-picking problem in 2 steps using mathematical programming.
- Simplifying the 2nd step of matheuristics excluding products that do not exist in the buffers from calculations. This approach was not present in Czerniachowska et al.'s (2023b) matheuristics.
- Simplifying the 2nd step of matheuristics excluding products with zero quantities, achieved in the 1st step of matheuristics. This approach was also not used in Czerniachowska et al. (2023b) matheuristics.

Conducting the computational experiments for 3 variants of the DC layouts, which differ with pickers to buffers assignment.

The contributions of the order-picking model proposed in this research which differ it from other order-picking models are the following:

- The racks with the products are grouped into buffers.
- The buffers are assigned to one or more pickers.
- The pickers are assigned to one or more buffers.
- The same product could be stored in different buffers, but the same product is allowed to be picked only from one buffer to the order.
- Containers are moved via the conveyor in one direction in a distribution center.
- A sequence of products that are picked to the container is important.

The order picking problem (OPP) investigated in this research is similar to the job shop scheduling problem (JSSP). Table 1 presents the comparison of OPP with JSSP.

Table 1
Comparison of OPP with JSSP

OPP	JSSP
Order	Job
Product	Operation
Picker	Machine
Travelling and picking time	Processing time
Sequence of picking products in the order	Sequence of operations in the job

The remainder of this paper is organized as follows. Related literature is discussed in the following section. After that, an order-picking problem definition is given. Its mathematical formulation is given in the next section. The following section explains the developed matheuristics. The next section reports the computational results. Finally, conclusions and an overview of future research opportunities are provided.

Related literature

Order picking takes up the greatest time of all the processes in a warehouse (Roodbergen and de Koster, 2001; de Koster et al., 2007) and costs between 55 and 75 percent of all warehousing expenses (Chiang et al., 2011; de Koster et al., 2007).

Order picking is the process of gathering items at the logistics hub. Creating a succession of plans for picking items from the shelves, choosing items, and distributing orders to workers is known as an order-picking problem (Iwasaki et al., 2013).

Order picking has grown in importance as a result of the increased focus on efficiency in logistics warehousing, and it is frequently suggested to use a range of various algorithms to reduce picking times (Wu et al., 2020).

Numerous papers examining order-picking methods have been published over the past few decades.

Several kinds of literature for order picking are surveyed and reviewed by De Koster et al. (2007). By emphasizing the most effective internal layout design, storage assignment methods, routing methods, order batching, and zoning, they organized typical decision problems in the design and management of order-picking processes. They included examples of class-based storage, zoning, batching, and routing techniques (De Koster et al., 2007).

Enhancing picking systems and customer service both depend on technological advancements and increased warehouse productivity. There are various picking system techniques that can be carried out by automated systems or people. Pinto et al. (2023) presented the theoretical foundation for the most popular picking system classifications among researchers. They focused on picking systems based on automation strategies and processes and picking systems based on picking optimization policies and procedures. They also showed that researchers should concentrate on methods that integrate people, warehouses, technology advancements, and the adoption of multi-objective algorithms.

The study by Iwasaki et al. (2013) suggested an approach to the order-picking issue that takes into account workers being crowded onto the same shelf at the logistics center. The proposed approach optimized the job-shop scheduling problem. Iwasaki et al. (2013) formulated the logistics center worker scheduling issue (Iwasaki et al., 2013).

Liu (1999) examined the location of stock and order picking in a distribution center where elements of dependent customer demands are presented. In order to optimize the stock location and picking procedure, clustering methods are used to extract the related information from the customer orders (Liu, 1999).

Roundy et al. (2005) created a more straightforward explanation of the order acceptance problem. The discrete-time variant is written as an integer program (Roundy et al., 2005). Roundy et al. (2005) proposed simulated annealing, a genetic algorithm, and a linear programming-based heuristic (Roundy et al., 2005).

Lee and Murray (2019) investigated a unique method for warehouse order picking, which was motivated by current developments in mobile robotic technology. Their study specifically took into account two types of commercially accessible mobile robots: a picker, which could rapidly move every item from the pick list to the packing station, and a transporter, which could grab goods off a shelf. The pick, place, and transport vehicle routing problem was a novel one that aimed to reduce the amount of time it took to convey every item on a pick list to the packing station. To address three connected research concerns, a formulation for mixed integer linear programming was created (Lee and Murray, 2019).

Researchers have previously created a variety of mathematical models that assist warehouse managers in assigning products to shelf placements, restructuring incoming orders, and directing order pickers through the warehouse in an effort to lower the cost of order picking (Van Gils et al., 2018). Often, it is wise to modify planning processes to the warehouse's unique layout (Roodbergen et al., 2015).

Haouassi et al. (2022) examined the effects of separating the orders (assigning the order lines of order to multiple pickers). In order to tackle the problem, the researchers expanded the integrated orders batching, batch scheduling, and picker routing problem by enabling order splitting, and they suggested a route first-schedule second heuristic. The routing phase's heuristic clustered the orders and built choosing tours that extracted the orderlines of each cluster using a split-based method. The created tours were allocated to pickers during the scheduling phase using a concept known as constraint programming to ensure that the order deadlines were met (Haouassi et al., 2022).

Onal et al. (2023) provided three quick fixes for the explosive storage picking issue. The two other solutions are heuristics that look for a starting or seed bin, whereas the first option concentrates on shrinking the MIP's decision space.

Effective order batching can also increase the effectiveness of order retrieval, as indicated by Gu, Goetschalckx, and McGinnis (2007). Research on warehouse order batching focused on breaking down a list of orders (a pick list) into batches such that each batch may be obtained in a timely manner. As a result, the batch size was established depending on the time needed for each batch to be completed (Petersen, 2000; de Koster, et al., 2007). The approaches for order batching that have been developed via research require established routes and take into account predetermined layout configurations.

A batching method for products based on the queuing theory was proposed by Tang and Chew (1997), and the outcome was used for storage assignment. They also gave a rectangle-picking method some thought (Tang and Chew, 1997).

Order picking can be improved using the high-performance pick system. In order to fulfill requests, it gives operators quick and easy access to the many boxes

or totes. Because of its ergonomic form, it makes the preparation of several orders at a time easy and is perfect for high-turnover items. Kawęcki and Gola (2022) presented a pick performance system as an IT support utilized for order complementing in an exemplary organization. The proposed method improved employee dismissal and helped to maintain productivity levels among the workforce. The authors also examined the features of the order complementation procedure.

One of the most researched optimization issues from the invention of computers to the present is job shop scheduling. It can be expressed simply as a constraint satisfaction problem due to its combinatorial character (Cinar et al., 2017; Hsu and Liu, 2009; Da Col and Teppan, 2019; Da Col and Teppan, 2022).

In the job shop, scheduling a group of machines and jobs with an ordered series of operations makes up the optimization issue. The execution duration of each task on a certain machine is known; thus, there is the need to find the start time and assigned machine of each operation so that the complete production ends as fast as possible. It is essential that the sequence of steps is followed and that no two jobs are carried out on the same equipment at once.

Due to its straightforward formulation and challenging situations to solve optimally, the job shop scheduling problem has earned particular notoriety. The makespan, or the amount of time between the beginning of the first operation and the conclusion of the last one, is the most common optimization criterion. The issue is described as a collection of tasks that a group of machines must complete (Da Col and Teppan, 2019).

In the job shop scheduling problem, the input for each job includes a specified order of operation types and equipment that are typically varied (Nowicki and Smutnicki, 1996). Complex factories that generate a variety of items, each requiring a unique workflow, are real-world examples (Da Col and Teppan, 2022).

Each job consists of a series of tasks, each of which needs to be completed by a particular machine and has a set processing time. Every task has a specific sequence of steps that must be followed. A series of operations on every machine in which the ordering of the operations is respected and there is no time overlap between operations on the same machine is an acceptable solution to this problem (Da Col and Teppan, 2019).

Industrial companies often use a facility to work with both Make-to-Stock (MTS) and Make-to-Order (MTO) products. The main goal of the MTO systems is to maintain a defined amount of completed goods inventory. MTS systems are used to fulfill a customer's order during a specific period of time.

Peeters and van Ooijen (2020) reviewed and classified the hybrid make-to-stock/make-to-order production control systems, introducing a taxonomy of different types of such systems.

Danilczuk et al. (2022) presented the premises of a procedure and a main algorithm for manufacturing job scheduling in hybrid make-to-order/make-to-stock production systems. They developed a shop job scheduling approach that enabled both to select jobs to be produced with the help of a make-to-stock strategy.

Kuthambalayan and Bera (2020) investigated the integrated firm's capacity for decision-making. They also worked with the factors that favor semi-finished goods substitution and the mixed MTS/MTO strategy over MTO and MTS strategies. The authors proposed a two-stage stochastic MINLP model that incorporated decisions regarding marketing about guaranteed lead-time and the product-wise mode of manufacturing and inventory level of semi-finished/finished items. The market demands were fulfilled in the second phase.

Various scheduling problems could be solved with constraint programming (Sadeh and Fox, 1996; Da Col and Teppan, 2019, Czerniachowska et al., 2023c).

Problem definition

We consider the following order-picking problem. In the logistics center (warehouse or DC), there is a given number of orders that must be completed by pickers among racks with shelves allocated in different buffers. The container leaves the depot with a list of products that must be picked from different racks situated in different places in the logistics center. The pickers are assigned to the buffers. One picker can be assigned to one buffer, and one picker can be assigned to more than one buffer nearby. Several pickers can also be assigned to one buffer.

The orders are known in advance, and no new orders can be added after they are batched. It is assumed that the logistics center has enough products to complete all orders, and that out-of-stock does not occur. When a container is coming to the buffer, the worker picks up the specified number of products, measures, cuts the product if needed, temporarily packs, and puts it into the container. The packing at this stage is temporal because later, the quality control team will open all packed products, count, and check them before packing to shipment to the client. If the picker is busy, the container may wait in the buffer waiting area near the rack. When all specified products are collected in the container, the container returns to the depot via the conveyor. Only one conveyor exists, which moves the

containers in one direction around the logistics center. The stops of the container are possible only in buffers.

Products are stored on racks, and the racks are assigned to buffers. Generally, the product is stored in one buffer. But in some situations, it is possible to store the same product in several buffers. For example, if the product from different deliveries may differ by color, even if it is the same color (for example, paints, knitting, or serving thread), it is suggested to store new delivery in another location with the goal of not allowing the picker to place product from different deliveries in the same order. Sometimes, when large deliveries occur, there is not enough space to store the product in one location; therefore, such product could also be stored in another buffer.

The sequence in which the products must be added to the containers is required. This requirement is based on the product types where, for example, big products should be added to the container first, or the products with the valid period should be added as late as possible.

The time that the picker spends moving between the locations in the same or different buffers is not taken into account, as it is assumed that in the responsibility of one picker, the buffers are close enough, and the racks in the same buffers are short enough. Therefore, the moving time of pickers is neglected. However, the time of the container moving between buffers via the conveyor is taken into account. In order to get to the previous buffer, the container must execute round-trip without exiting from the final depot because the conveyor is one-directional.

After the orders are picked, they need to be checked by quality control, packed, and shipped to the customers. There are other stages of the picking process in the logistics center that are not investigated in this study. Only the order-picking step is modeled and investigated.

The time the container spends traveling from one buffer to another is the makespan time for the task. When all products in the orders are assigned to be picked, the longest cumulated traveling time of the container becomes the maximum makespan time for the OPP. The goal of the OPP is to find how to assign orders to be picked from the racks in the logistics center repeatedly and how to travel in the logistics center from one buffer to another so as to minimize the maximum makespan time.

In this research, we examine three layouts of a distribution center. An example with 6 buffers (B1-B6) is shown in Figures 1, 2, and 3. B0 is the start buffer, B7 is the final depot. The direction of moving the containers via the conveyor is from B0 to B7. There are 120 locations allocated in 6 buffers per 20 main locations in each buffer. There are also 8 additional locations in each buffer for the products which are stored in several buffers. In Figure 1, a picker completes orders in a single buffer. In Figure 2, a picker completes orders in several buffers; for example, Picker 4 moves between buffers B4 and B5. In Figure 3, several pickers complete orders in one buffer; for example, pickers 1, 2, and 5, 6 manage orders in buffers B1 and B6, respectively.

Problem formulation

Next, we present the mathematical model for the described problem. The following notation is used in the model.

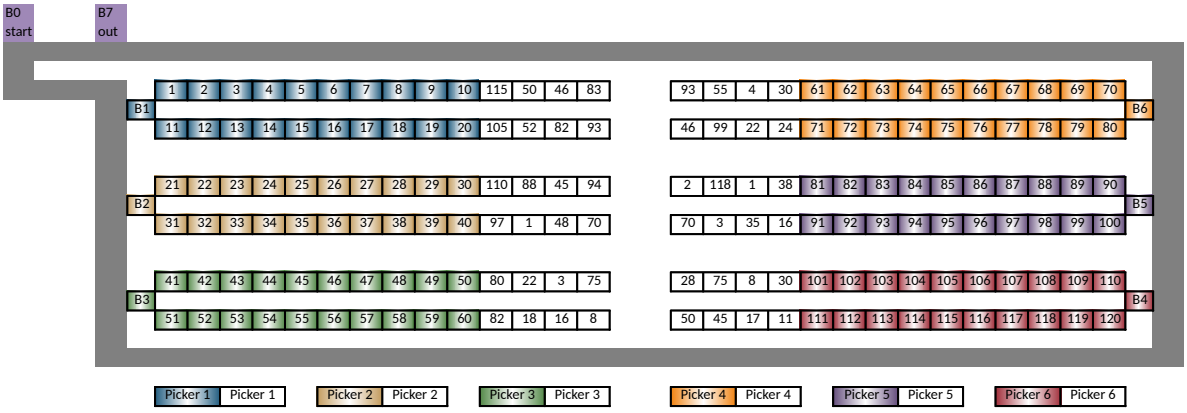

Fig. 1. The layout of a distribution center in which a picker completes orders in a single buffer: B0 – start buffer; B7 – out buffer; B1-B6 – order picking buffers; Dark colors in buffers B1-B6 – regular buffer products; Light colors in buffers B1-B6 – products also allocated in other buffers

Fig. 2. The layout of a distribution center in which a picker completes orders in several buffers: B0 – start buffer; B7 – out buffer; B1-B6 – order picking buffers; Dark colors in buffers B1-B6 – regular buffer products; Light colors in buffers B1-B6 – products also allocated in other buffers

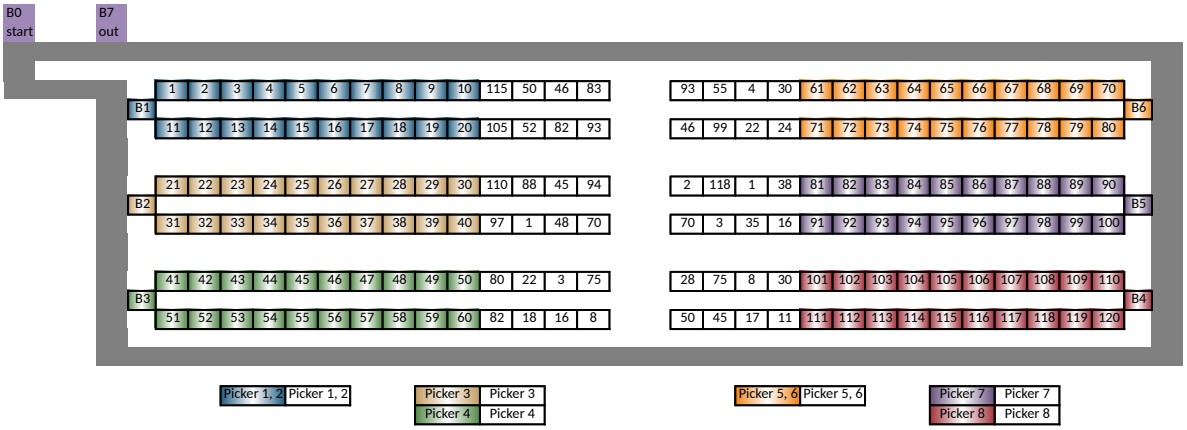

Fig. 3. The layout of a distribution center in which several pickers complete orders in one buffer: B0 – start buffer; B7 – out buffer; B1-B6 – order picking buffers; Dark colors in buffers B1-B6 – regular buffer products; Light colors in buffers B1-B6 – products also allocated in other buffers

Indices and sets:

- N – number of orders to be completed;
- H – number of products in the distribution center;
- R – number of buffers on the conveyor;
- W – number of pickers working on the conveyors;
- i, m – index for orders, $i, m = 1, \ldots, N$;
- j, k – index for products, $j, k = 1, \ldots, H$;
- b, g – index for buffers, $b, g = 1, \ldots, R$;
- p, l – index for pickers, $p = 1, \ldots, W$.

Parameters:

- t_{ij} – estimated picking time of the product j in the order i;
- d_{ij} – sequence of picking operation of the product j in the order i;
- s_b – container travel time from the depot to the buffer b;
- c_{ij} – ordered quantity of product j in the order i that must be picked by pickers;
- q_{bj} – available quantity of the product j in the buffer b;
- f_{bg} – container travel time from the buffer b to the buffer g (because of the fact that the conveyor is one-directional, $f_{bg} \neq f_{gb}$);
- z_{bj} – binary parameter of availability of products in buffers;

$$z_{bj} = \left\{ \begin{array}{l} 1, \text{ if the product } j \text{ is located} \\ \text{in the buffer } b \\ 0, \text{ otherwise} \end{array} \right\};$$

v_{pb} – binary parameter of assignment pickers to buffers in the distribution center;

$$v_{pb} = \left\{ \begin{array}{l} 1, \text{ if the picker } p \text{ is assigned} \\ \text{to the buffer } b \\ 0, \text{ otherwise} \end{array} \right\};$$

Decision variables:

x_{ijpb} – start time of the processing of the product j in the order i processed by the picker p in the buffer b;

y_{ijpb} – quantity of the product j in the order i processed by the picker p in the buffer b.

Decision expressions:

t_p^{pkr} – total processing time of the picker p;

t_b^{bfr} – total time of products processing in the buffer b.

Minimize the makespan:

$$\min \max_{i=1,\ldots,N} \max_{\substack{j=1,\ldots,H \\ t_{ij}\neq 0}} \max_{p=1,\ldots,W} \max_{b=1,\ldots,R} (x_{ijpb} + t_{ij}) \quad (1)$$

Subject to:

The picker gathers products only in the buffers to which the picker is assigned.

$$\forall(p)\forall(i)\forall(j)\forall(b : v_{pb} = 0)[y_{ijpb} = 0] \quad (2)$$

The picker gathers products only in the buffers in which the product is located.

$$\forall(p)\forall(i)\forall(j)\forall(b : z_{bj} = 0)[y_{ijpb} = 0] \quad (3)$$

Each product for the order is processed in one buffer by one picker.

$$\forall(i)\forall(j)\left[\sum_{p=1}^{W}\sum_{b=1}^{R} \min(y_{ijpb}, 1) \leq 1\right] \quad (4)$$

Enough products exist in each buffer to be gathered by all pickers for each separate order.

$$\forall(i)\forall(j)\forall(b)\left[\sum_{p=1}^{W} y_{ijpb} \leq q_{bj}\right] \quad (5)$$

Enough quantity of products are gathered.

$$\forall(i)\forall(j)\left[\sum_{p=1}^{W}\sum_{b=1}^{R} y_{ijpb} = c_{ij}\right] \quad (6)$$

Exclude not-ordered products from calculations.

$$\forall(i)\forall(j : t_{ij} = 0)\forall(p)\forall(b)[x_{ijpb} = 0] \quad (7)$$

Each picker works with one product, which is placed for one order in all of the buffers to which the picker is assigned at a time.

$$\begin{array}{l} \forall(p)\forall(i,m : i \neq m)\forall(b : v_{pb} = 1) \\ \forall(j : t_{ij} \neq 0, t_{mj} \neq 0) \\ [x_{ijpb} + t_{ij} \leq x_{mjpb} \vee x_{mjpb} + t_{mj} \leq x_{ijpb}] \end{array} \quad (8)$$

Each picker in each order in all the buffers to which the picker is assigned works with one product at a time.

$$\begin{array}{l} \forall(p)\forall(i)\forall(b : v_{pb} = 1) \\ \forall(j,k : j \neq k, t_{ij} \neq 0, t_{ik} \neq 0) \\ [x_{ijpb} + t_{ij} \leq x_{ikpb} \vee x_{ikpb} + t_{ik} \leq x_{ijpb}] \end{array} \quad (9)$$

Each picker in each order for all products works with products at one of the buffers the picker is assigned at a time.

$$\begin{array}{l} \forall(p)\forall(i)\forall(j : t_{ij} \neq 0) \\ \forall(b,g : b \neq g, v_{pb} = 1, v_{pg} = 1) \\ [x_{ijpb} + t_{ij} \leq x_{ijpg} \vee x_{ijpg} + t_{ij} \leq x_{ijpb}] \end{array} \quad (10)$$

Each picker at each of the buffers to which the picker is assigned works with one of the different orders and with one of the different products at a time.

$$\begin{array}{l} \forall(p)\forall(b : v_{pb} = 1)\forall(i,m : i \neq m) \\ \forall(j,k : j \neq k, t_{ij} \neq 0, t_{mk} \neq 0) \\ [x_{ijpb} + t_{ij} \leq x_{mkpb} \vee x_{mkpb} + t_{mk} \leq x_{ijpb}] \end{array} \quad (11)$$

Each picker works with one of the different orders, with one of the different products, at one of the buffers the picker is assigned at a time.

$$\begin{array}{l} \forall(p)\forall(b,g : b \neq g, v_{pb} = 1, v_{pg} = 1) \\ \forall(i,m : i \neq m) \\ \forall(j,k : j \neq k, t_{ij} \neq 0, t_{mk} \neq 0) \\ [x_{ijpb} + t_{ij} \leq x_{mkpg} \vee x_{mkpg} + t_{mk} \leq x_{ijpb}] \end{array} \quad (12)$$

The start of the picking operation must include the travel time from the depot to the current buffer.

$$\begin{array}{l} \forall(b)\forall(j : z_{bj} = 1)\forall(i : t_{ij} \neq 0)\forall(p) \\ [x_{ijpb} \geqslant z_{bj} s_b] \end{array} \quad (13)$$

Operations precedence in each job considering travel time between buffers, i.e., the travel time of the container between buffers must be enough (considering traveling in one direction).

$$\forall(i)\forall(j,k: j \neq k, t_{ij} \neq 0, t_{ik} \neq 0, d_{ij}+1 = d_{ik})$$
$$\forall(b,g: z_{bj}=1, z_{gk}=1)\forall(p,l: v_{pb}=1, v_{lg}=1)$$
$$[x_{ijpb} + t_{ij} + \min(y_{ijpb},1)\cdot\min(y_{iklg},1)\cdot r_{bg} \leq x_{iklg}] \tag{14}$$

Picking operations precedence in each order, executed by all pickers in all buffers, i.e., the previous picking operation must finish before the start of any next picking operation for this order.

$$\forall(i)\forall(j,k: j \neq k, t_{ij} \neq 0, t_{ik} \neq 0)$$
$$\left[\max_{p=1,\ldots,W}\max_{b=1,\ldots,R}(x_{ijpb}+t_{ij}) \leq \min_{p=1,\ldots,W}\min_{b=1,\ldots,R} x_{ikpb} \vee \vee \max_{p=1,\ldots,W}\max_{b=1,\ldots,R}(x_{ikpb}+t_{ik}) \leq \min_{p=1,\ldots,W}\min_{b=1,\ldots,R} x_{ijpb}\right] \tag{15}$$

Picking operations precedence according to the sequence of picking operations.

$$\forall(i)\forall(j,k: j \neq k, t_{ij} \neq 0, t_{ik} \neq 0, d_{ij}+1 = d_{ik})$$
$$\forall(p)\forall(b)[x_{ijpb} + t_{ij} \leq x_{ikpb}] \tag{16}$$

Decision variables:
Start time of the container processing

$$x_{ijpb} = \{0, \mathbb{Z}^+\} \tag{17}$$

Quantity of the gathered products

$$y_{ijpb} = \{0, \mathbb{Z}^+\} \tag{18}$$

Solving the problem with matheuristics

In the real-life world of the field of computing, scheduling is a decision-making process. Given the limited resources available, creating a schedule might be challenging. One of the mainstays of contemporary living is e-commerce. Order picking scheduling plays an important role in handling high volatility in warehouses and distribution centers.

Heuristics are a set of guidelines that help to focus an investigation. Even if they cannot guarantee the optimum solution, they may typically do it fast and with minimal effort. When more traditional methods are ineffective, a heuristic is a technique that is intended to solve a problem more rapidly. Various scenarios call for particular heuristics.

Matheuristics are optimization algorithms that generate heuristic solutions while being independent of the underlying optimization problems. Although heuristic and mathematics algorithms are frequently easy to set up and run quickly, they run the risk of missing superior solutions or getting caught in local optima. Heuristics and matheuristics can be avoided by making decisions carefully rather than automatically.

Combining mathematical programming with heuristic and metaheuristic methods is a viable form of hybridization. Some algorithms heavily rely on features that are obtained from the mathematical model of the optimization issues. Heuristics with elements of mathematical programming typically are developed because they are effective methods that allow people to react to decisions or challenges as well.

Regarding scheduling issues, commercial solvers can sometimes provide the best solution or even none at all, but not always. Various solvers optimize schedules with constraints. Such an approach reduces costs and improves service quality. Matheuristics can be developed using solvers.

Since it would be difficult to develop a method that could generate high-quality solutions in an acceptable period of time while taking into account all order-picking constraints obtained from the distribution centers and warehouses, a mathematical programming-based approach was chosen for this research. To save the effort, we suggest breaking the primary order picking task down into two more manageable problems and attempting to resolve each one separately using mathematical programming. We propose the variants of matheuristics, each of which consists of two steps. The first step differs depending on the matheuristics variant (Figure 4). The second step is the same for both variants (Figure 5). There are two variations of the first step: the Pickers free time minimization problem and the Buffers free time minimization problem.

Figure 4 presents the first step of the main order picking makespan minimization problem decomposition and solving Pickers and Buffers free time mini-

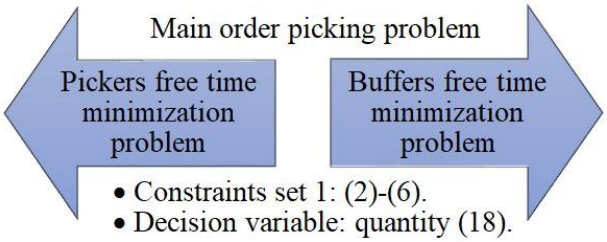

Fig. 4. The 1st step of the 2 variants of matheuristics

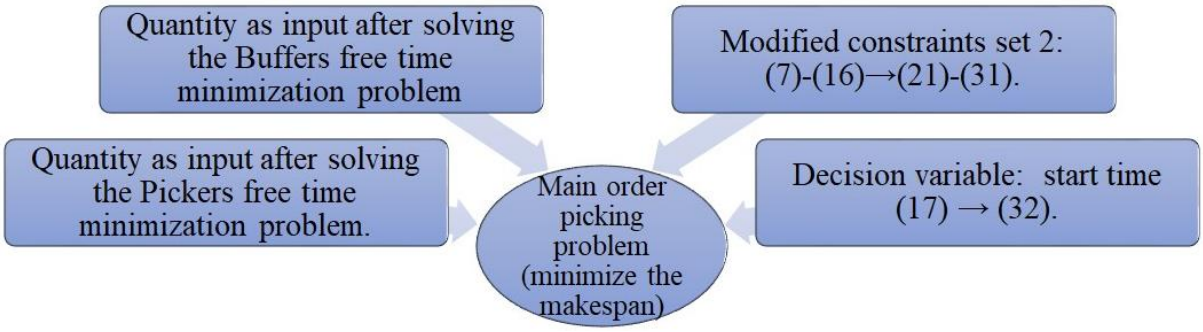

Fig. 5. The 2nd step of both matheuristics

mization problems using the same set of constraints (2)-(6) and the same decision variable (18). The result of the solution Pickers and Buffers free time minimization problems is product quantity. We next use it as input for the second step and solve the primary Order picking makespan minimization problem (Figure 5). Here also, for both matheuristics we have the same set of constraints (7)-(16) and the same decision variable (17). We transform the mentioned constraints set into the (21)-(31), in which we exclude products with zero quantities achieved in the previous step. This method significantly simplifies calculations. The decision variable (32) is the same as (17). The result of the solution is the container's start time.

Let's provide additional decision expressions.

Decision expressions:

$$t_p^{pkr} = \sum_{i=1}^{N}\sum_{j=1}^{H}\sum_{b=1}^{R} \min(y_{ijpb}, 1) \cdot t_{ij}$$

$$t_b^{bfr} = \sum_{i=1}^{N}\sum_{j=1}^{H}\sum_{p=1}^{W} \min(y_{ijpb}, 1) \cdot t_{ij}$$

The criteria function for matheuristics are:

Matheuristics 1: Minimize the pickers' free time.

$$\min \sum_{p=1}^{W} \left(\max_{l=1,\ldots,W} t_l^{pkr} - t_p^{pkr} \right) \tag{19}$$

Matheuristics 2: Minimize the buffers' free time.

$$\min \sum_{b=1}^{R} \left(\max_{g=1,\ldots,R} t_g^{bfr} - t_b^{bfr} \right) \tag{20}$$

Subject to:

Exclude not-ordered products from calculations.

$$\forall(i)\forall(j : t_{ij} = 0)\forall(p)\forall(b)[x_{ijpb} = 0] \tag{21}$$

Exclude products that do not exist in buffer from calculations.

$$\forall(i)\forall(j)\forall(p)\forall(b : y_{ijpb} = 0)[x_{ijpb} = 0] \tag{22}$$

Each picker works with one product, which is placed for one order in all of the buffers to which the picker is assigned at a time.

$$\begin{aligned}&\forall(p)\forall(i, m : i \neq m)\forall(b : v_{pb} = 1)\\ &\forall(j : t_{ij} \neq 0, t_{mj} \neq 0, y_{ijpb} \neq 0, y_{mjpb} \neq 0)\\ &[x_{ijpb} + t_{ij} \leq x_{mjpb} \vee x_{mjpb} + t_{mj} \leq x_{ijpb}]\end{aligned} \tag{23}$$

Each picker in each order in all the buffers to which the picker is assigned works with one product at a time.

$$\begin{aligned}&\forall(p)\forall(i)\forall(b : v_{pb} = 1)\\ &\forall(j, k : j \neq k, t_{ij} \neq 0, t_{ik} \neq 0,\\ &y_{ijpb} \neq 0, y_{ikpb} \neq 0)\\ &[x_{ijpb} + t_{ij} \leq x_{ikpb} \vee x_{ikpb} + t_{ik} \leq x_{ijpb}]\end{aligned} \tag{24}$$

Each picker in each order for all products works with products at one of the buffers the picker is assigned at a time.

$$\begin{aligned}&\forall(p)\forall(i)\forall(j : t_{ij} \neq 0)\\ &\forall(b, g : b \neq g, v_{pb} = 1, v_{pg} = 1,\\ &y_{ijpb} \neq 0, y_{ijpg} \neq 0)\\ &[x_{ijpb} + t_{ij} \leq x_{ijpg} \vee x_{ijpg} + t_{ij} \leq x_{ijpb}]\end{aligned} \tag{25}$$

Each picker at each of the buffers to which the picker is assigned works with one of the different orders and with one of the different products at a time.

$$\begin{aligned}&\forall(p)\forall(b : v_{pb} = 1)\forall(i, m : i \neq m)\\ &\forall(j, k : j \neq k, t_{ij} \neq 0, t_{mk} \neq 0,\\ &y_{ijpb} \neq 0, y_{mkpb} \neq 0)\\ &[x_{ijpb} + t_{ij} \leq x_{mkpb} \vee x_{mkpb} + t_{mk} \leq x_{ijpb}]\end{aligned} \tag{26}$$

Each picker works with one of the different orders, with one of the different products, at one of the buffers the picker is assigned at a time.

$$\forall(p)\forall(b,g:b\neq g,v_{pb}=1,v_{pg}=1)$$
$$\forall(i,m:i\neq m)$$
$$\forall(j,k:j\neq k,t_{ij}\neq 0,t_{mk}\neq 0,$$
$$y_{ijpb}\neq 0,y_{mkpg}\neq 0,y_{ijpg}\neq 0,y_{mkpb}\neq 0)$$
$$[x_{ijpb}+t_{ij}\leq x_{mkpg}\vee x_{mkpg}+t_{mk}\leq x_{ijpb}] \quad (27)$$

The start of the picking operation must include the travel time from the depot to the current buffer.

$$\forall(b)\forall(j:z_{bj}=1)\forall(i:t_{ij}\neq 0)$$
$$\forall(p:y_{ijpb}\neq 0)[x_{ijpb}\geqslant z_{bj}s_b] \quad (28)$$

Operations precedence in each job considering travel time between buffers, i.e., the travel time of the container between buffers must be enough (considering traveling in one direction).

$$\forall(i)\forall(j,k:j\neq k,t_{ij}\neq 0,t_{ik}\neq 0,d_{ij}+1=d_{ik})$$
$$\forall(b,g:z_{bj}=1,z_{gk}=1)$$
$$\forall(p,l:v_{pb}=1,v_{lg}=1,y_{ijpb}\neq 0,y_{iklg}\neq 0)$$
$$[x_{ijpb}+t_{ij}+r_{bg}\leq x_{iklg}] \quad (29)$$

Picking operations precedence in each order, executed by all pickers in all buffers, i.e., the previous picking operation must finish before the start of any next picking operation for this order.

$$\forall(i)\forall(j,k:j\neq k,t_{ij}\neq 0,t_{ik}\neq 0)$$
$$\Big[\max_{p=1,\ldots,W}\max_{\substack{b=1,\ldots,R,\\ y_{ijpb}\neq 0}}(x_{ijpb}+t_{ij})$$
$$\leq \min_{p=1,\ldots,W}\min_{\substack{b=1,\ldots,R,\\ y_{ikpb}\neq 0}}x_{ikpb}$$
$$\vee \max_{p=1,\ldots,W}\max_{\substack{b=1,\ldots,R,\\ y_{ikpb}\neq 0}}(x_{ikpb}+t_{ik})$$
$$\leq \min_{p=1,\ldots,W}\min_{\substack{b=1,\ldots,R,\\ y_{ijpb}\neq 0}}x_{ijpb}\Big] \quad (30)$$

Picking operations precedence according to the sequence of picking operations.

$$\forall(i)\forall(j,k:j\neq k,t_{ij}\neq 0,t_{ik}\neq 0,d_{ij}+1=d_{ik})$$
$$\forall(p)\forall(b:y_{ijpb}\neq 0,y_{ikpb}\neq 0)$$
$$[x_{ijpb}+t_{ij}\leq x_{ikpb}] \quad (31)$$

Decision variable:

Start time of the container processing

$$x_{ijpb}=\{0,\mathbb{Z}^+\} \quad (32)$$

Experiment

The computational experiment simulates the DC realities and presents the quality of the solution obtained by the developed matheuristics. The results obtained during the experiment will provide evidence of matheuristics applicability to solve real order-picking problems in the DC. During the experiment, we also tested the effectiveness of the developed model, providing variants of the DC layout.

The experiments were performed using the commercial solver IBM ILOG CPLEX Optimization Studio Version: 12.10.0.0. The computer parameters were: Processor: AMD Ryzen 5 1600 Six-Core Processor 3.20 GHz; System type: 64-bit Operation System, x64-based processor; RAM: 16 GB; Operation system: Windows 10.

For the experiment, 120 instances were generated – 10 tests for each setting set. There were 3 DC layouts with 6 buffers in each one. There was a different number of pickers in each layout: 6 pickers in the 1st layout, 4 pickers in the 2nd layout, 8 pickers in the 3rd layout. There were 5, 10, 15, 20 orders for each of the layouts. In each order, the container might visit from 1 to 5 locations to pick items in the logistics center layout. In the buffers on each layout, there were 120 locations for regular products and 48 locations for repeated inside other buffers products in the logistics center.

The picking time varied from 1 to 30 minutes. A number of products that might be picked in the order and picking time were selected randomly. Order picking is the longest operation while completing the order. The picker must find the product on the shelf, count an appropriate number of items of product, measure and cut the appropriate length of material, temporarily pack it, and put it into the container. Therefore, the picking time is so varied.

The product is stored in the DC in 1, 2, or 3 locations. 70.83% of products (85 items) are stored only in one location. 18.33% of products (22 items) are stored in two locations. 10.83% of products (13 items) are stored in three locations. The time limit for CPLEX was set to 5 minutes.

Figure 6 presents the comparison of the 2-step matheuristics and Full model solution for 5 orders where 2-step matheuristics were better than Full model solution. For the 1st layout L1, both Pickers and Buffers matheuristics found the same solutions. Matheuristics were better in 2 of 10 instances on average at 11.61% compared to the Full model solution.

For the 2nd layout L2, Pickers and Buffers matheuristics found the same solutions for 5 of 10 instances. Pickers matheuristics were better than Buffers matheuris-

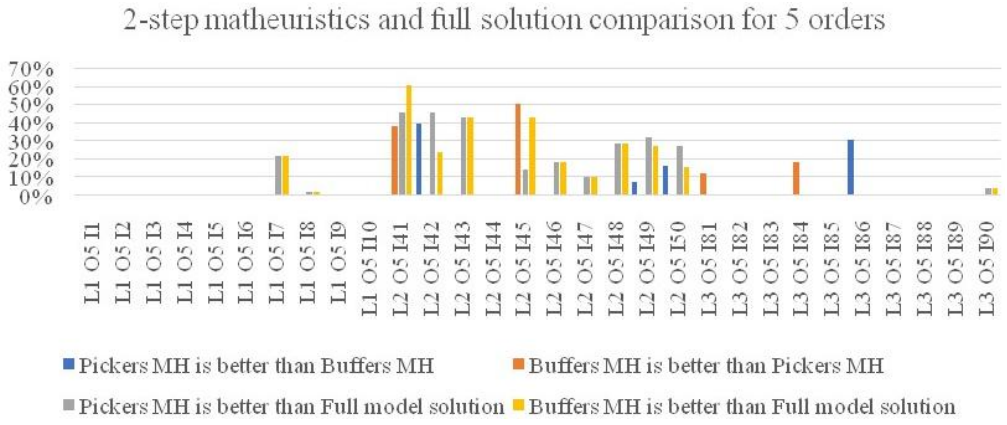

Fig. 6. Comparison of the 2-step Pickers, Buffers matheuristics and Full model solution for 5 orders: L{1,2,3} – layouts, O5–5 orders, I {1..10, 41..50, 81–90} – instances

tics in 3 instances, on average at 20.86%, while Buffers matheuristics were better than Pickers matheuristics in 2 instances, on average at 44.15%. Compared to the Full model solution, matheuristics were better in 9 of 10 instances. Pickers matheuristics were better on average at 29.47%. Buffers matheuristics were better on average at 30.09%.

For the 3rd layout L3, Pickers and Buffers matheuristics found the same solutions for 6 of 10 instances. Pickers matheuristics were better than Buffers matheuristics only in 1 instance at 30.69%, while Buffers matheuristics were better than Pickers matheuristics in 3 instances on average at 9.92%. Compared to the Full model solution, matheuristics were better only in 1 instance at 4.28%. Therefore, matheuristics could be mostly applicable for the 2nd layout L2.

Figure 7 presents the comparison of the 2-step matheuristics and Full model solution for 10 orders. For the 1st layout L1, Pickers and Buffers matheuristics found the same solutions for 9 of 10 instances. Buffers matheuristics were better than Pickers matheuristics in 1 instance at 1.67%. Compared to the Full model solution, matheuristics were better in 1 of 10 instances at 2.08%.

For the 2nd layout L2, Pickers and Buffers matheuristics found the same solutions for 9 of 10 instances. Buffers matheuristics were better than Pickers matheuristics in 1 instance at 6.41%. Compared to the Full model solution, matheuristics were better in all 10 instances. Pickers matheuristics were better on average at 36.33%. Buffers matheuristics were better on average at 36.80%.

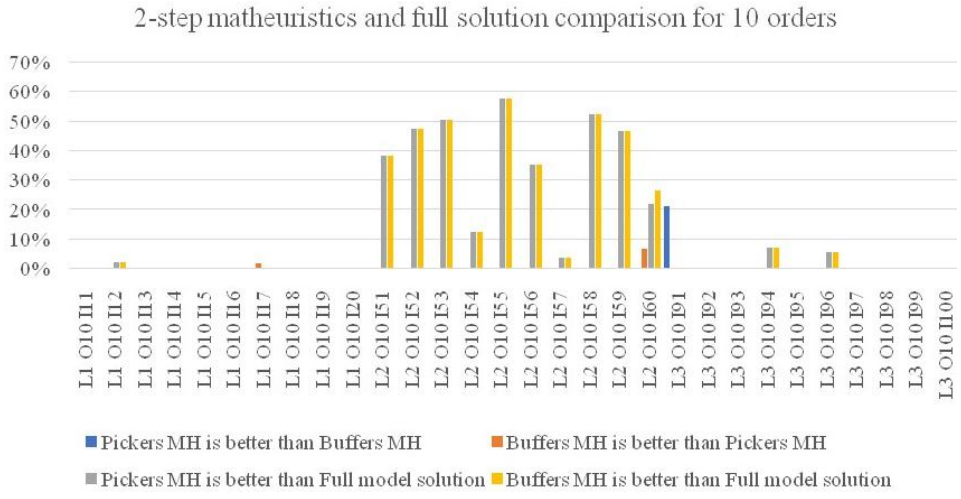

Fig. 7. Comparison of the 2-step Pickers, Buffers matheuristics and Full model solution for 10 orders: L{1,2,3} – layouts, O10–10 orders, I {11..20, 51..60, 91–100} – instances

For the 3rd layout L3, Pickers and Buffers matheuristics found the same solutions for 9 of 10 instances. Pickers matheuristics were better than Buffers matheuristics only in 1 instance at 20.78%. Compared to the Full model solution, matheuristics were better in 2 instances on average at 5.95%. Therefore, matheuristics could be mostly applicable for the 2nd layout L2.

Figure 8 presents the comparison of the 2-step matheuristics and Full model solution for 15 orders. For the 1st layout L1, Pickers and Buffers matheuristics found the same solutions for 8 of 10 instances. Buffers matheuristics were better than Pickers matheuristics in 2 instances on average at 11.76%. Compared to the Full model solution, matheuristics were better in all 10 instances. Pickers matheuristics were better on average at 45.33%. Buffers matheuristics were better on average at 46.57%. There were no cases where the Full model solution was better than matheuristics.

For the 2nd layout L2, Pickers and Buffers matheuristics found the same solutions only for 2 of 10 instances. Pickers matheuristics were better than Buffers matheuristics in 6 instances on average at 9.40%, while Buffers matheuristics were better than Pickers matheuristics in 2 instances on average at 14.06%. Compared to the Full model solution, matheuristics were better in all 10 instances. Pickers matheuristics were better on average at 69.13%. Buffers matheuristics were better on average at 68.18%. There were no cases where the Full model solution was better than matheuristics.

For the 3rd layout L3, Pickers and Buffers matheuristics found the same solutions for 5 of 10 instances. Pickers matheuristics were better than Buffers matheuristics only in 2 instances on average at 8.37%, while Buffers matheuristics were better than Pickers matheuristics in 3 instances on average at 22.31%. Compared to the Full model solution, matheuristics were better in all 10 instances. Pickers matheuristics were better on average at 31.73%. Buffers matheuristics were better on average at 34.64%. There were no cases where the Full model solution was better than matheuristics.

Therefore, matheuristics could be successfully applicable for all layouts.

Figure 9 presents the comparison of the 2-step matheuristics and Full model solution for 20 orders.

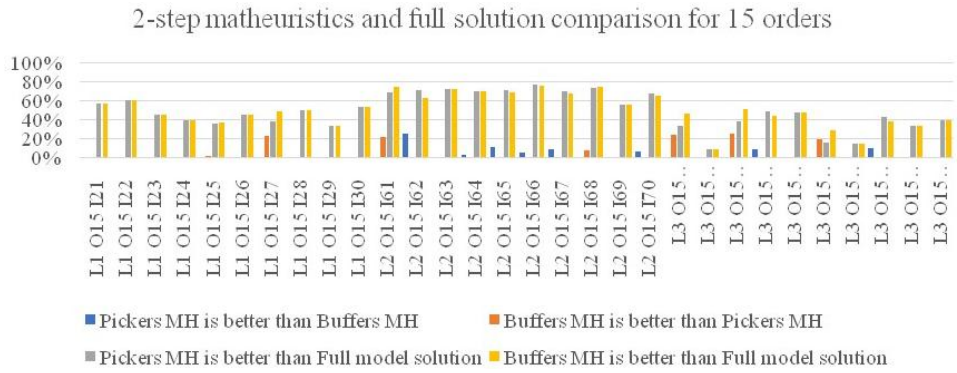

Fig. 8. Comparison of the 2-step Pickers, Buffers matheuristics and Full model solution for 15 orders: L{1,2,3} – layouts, O15–15 orders, I {21..30, 61..70, 101–110} – instances

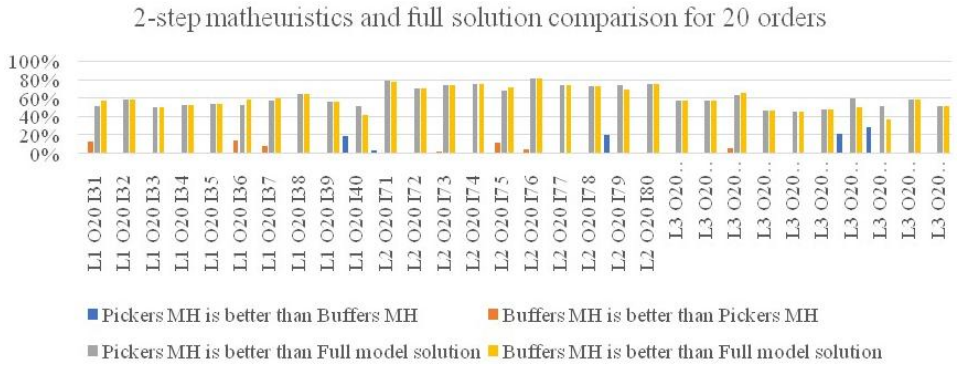

Fig. 9. Comparison of the 2-step Pickers, Buffers matheuristics and Full model solution for 20 orders: L{1,2,3} – layouts, O20–20 orders, I {31..40, 71..80, 111–120} – instances

For the 1st layout L1, Pickers and Buffers matheuristics found the same solutions for 5 of 10 instances. Pickers matheuristics were better than Buffers matheuristics only in 1 instance at 19.34%. Buffers matheuristics were better than Pickers matheuristics in 4 instances on average at 9.36%. Compared to the Full model solution, matheuristics were better in all 10 instances. Pickers matheuristics were better on average at 54.48%. Buffers matheuristics were better on average at 55.10%. There were no cases where the Full model solution was better than matheuristics.

For the 2nd layout L2, Pickers and Buffers matheuristics found the same solutions for 5 of 10 instances. Pickers matheuristics were better than Buffers matheuristics in 2 instances on average at 12.13%, while Buffers matheuristics were better than Pickers matheuristics in 3 instances on average at 6.28%. Compared with the Full model solution, matheuristics were better in all 10 instances. Pickers matheuristics were better on average at 74.28%. Buffers matheuristics were better on average at 74.15%. There were no cases where the Full model solution was better than matheuristics.

For the 3rd layout L3, Pickers and Buffers matheuristics found the same solutions for 7 of 10 instances. Pickers matheuristics were better than Buffers matheuristics only in 2 instances on average at 25.09%, while Buffers matheuristics were better than Pickers matheuristics only in 1 instance at 6.49%. Compared to the Full model solution, matheuristics were better in all 10 instances. Pickers matheuristics were better on average at 53.78%. Buffers matheuristics were better on average at 51.73%. There were no cases where the Full model solution was better than matheuristics.

Therefore, matheuristics could be successfully applicable for all layouts.

Figure 10 presents the comparison of the Full model solution and 2-step matheuristics for 5 orders where the Full model solution was better than 2-step matheuristics. For the 1st layout L1, the Full model solution was better in 2 of 10 instances on average at 15.51%. For the 2nd layout L2, there were no cases where the Full model solution was better than matheuristics. For the 3rd layout L3, the Full model solution was better than Pickers matheuristics in 7 instances on average at 14.41%. The Full model solution was better than Buffers matheuristics in 6 instances on average at 15.77%.

Figure 11 presents the comparison of the Full model solution and2-step matheuristics for 10 orders where the Full model solution was better than 2-step matheuristics. For the 1st layout L1, the Full model solution was better than matheuristics in 6 instances. The Full model solution was better than Pickers matheuristics on average at 10.63%. The Full model solution was better than Buffers matheuristics on average at 10.36%.For the 2nd layout L2, there were no cases where the Full model solution was better than matheuristics. For the 3rd layout L3, the Full model solution was better than Pickers matheuristics in 3 instances on average at 16.33%. The Full model solution was better than Buffers matheuristics in 4 instances on average at 16.55%.

There is no comparison of the Full model solution and 2-step matheuristics for 15 and 20 orders because, for all cases, the Full model solution was worse than 2-step matheuristics. As it could be observed, with an increasing number of orders, the quality of the Full model solution is decreased, and matheuristics should be applied.

Figure 12 presents the 2-step matheuristics execution speed comparison for 5 orders. For the 1st layout L1, Pickers matheuristics were faster than Buffers matheuristics in 4 instances on average at 8.82%, while Buffers matheuristics were faster than Pickers

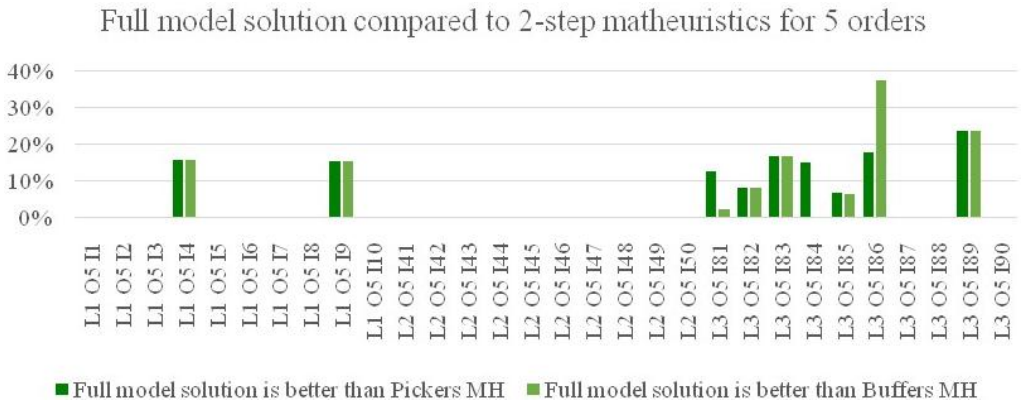

Fig. 10. Full model solution compared to 2-step Pickers, Buffers matheuristics where Full model solution was better for 5 orders: L{1,2,3} – layouts, O5–5 orders, I {1..10, 41..50, 81–90} – instances

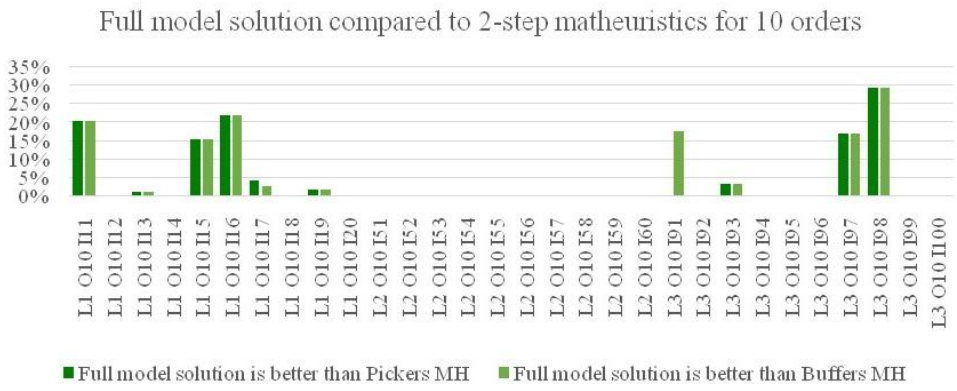

Fig. 11. Full model solution compared to 2-step Pickers, Buffers matheuristics where Full model solution was better for 10 orders: L{1,2,3} – layouts, O10–10 orders, I {11..20, 51..60, 91–100} – instances

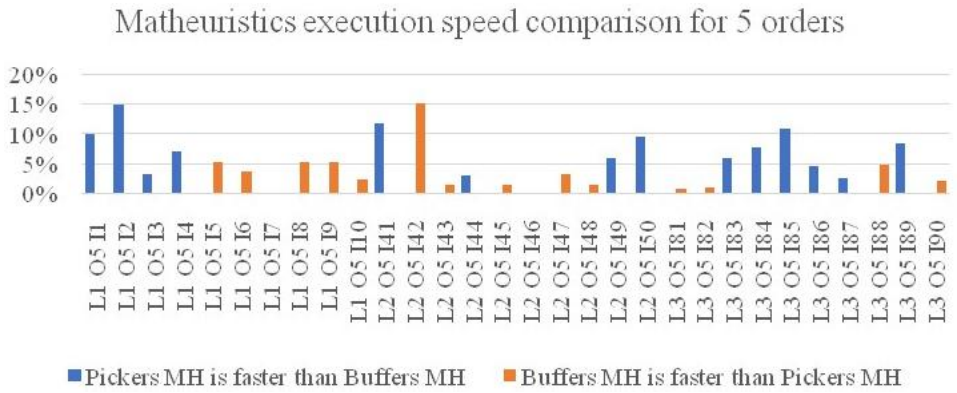

Fig. 12. 2-step Pickers, Buffers matheuristics execution speed comparison for 5 orders: L{1,2,3} – layouts, O5–5 orders, I {1..10, 41..50, 81–90} – instances

matheuristics in 5 instances on average at 4.44%. For the 2nd layout L2, Pickers matheuristics were faster than Buffers matheuristics in 4 instances on average at 7.62%, while Buffers matheuristics were faster than Pickers matheuristics in 5 instances on average at 4.59%. For the 3rd layout L3, Pickers matheuristics were faster than Buffers matheuristics in 6 instances on average at 6.75%, while Buffers matheuristics were faster than Pickers matheuristics in 4 instances on average at 2.25%.

For the 1st layout, the solution time for Pickers matheuristics was, on average, 0.83 seconds. For the 1st layout, the solution time for Buffers matheuristics was, on average, 0.85 seconds. For the 2nd layout, the solution time for Pickers matheuristics was, on average, 0.63 seconds. For the 2nd layout, the solution time for Buffers matheuristics was, on average, 0.64 seconds. For the 3rd layout, the solution time for Pickers matheuristics was, on average, 1.09 seconds. For the 3rd layout, the solution time for Buffers matheuristics was, on average, 1.13 seconds.

Figure 13 presents the 2-step matheuristics execution speed comparison for 10 orders. For the 1st layout L1, Pickers matheuristics were faster than Buffers matheuristics in 5 instances on average at 4.69%, while Buffers matheuristics were faster than Pickers matheuristics in 5 instances on average at 5.11%. For the 2nd layout L2, Pickers matheuristics were faster than Buffers matheuristics in 5 instances on average at 3.45%, while Buffers matheuristics were faster than Pickers matheuristics in 5 instances on average at 5.98%.For the 3rd layout L3, Pickers matheuristics were faster than Buffers matheuristics in 6 instances on average at 8.20%, while Buffers matheuristics were faster than Pickers matheuristics in 4 instances on average at 2.88%.

For the 1st layout, the solution time for Pickers matheuristics was, on average, 1.83 seconds. For the 1st layout, the solution time for Buffers matheuristics was, on average, 1.82 seconds. For the 2nd layout, the solution time for Pickers matheuristics was, on average, 1.53 seconds. For the 2nd layout, the solution

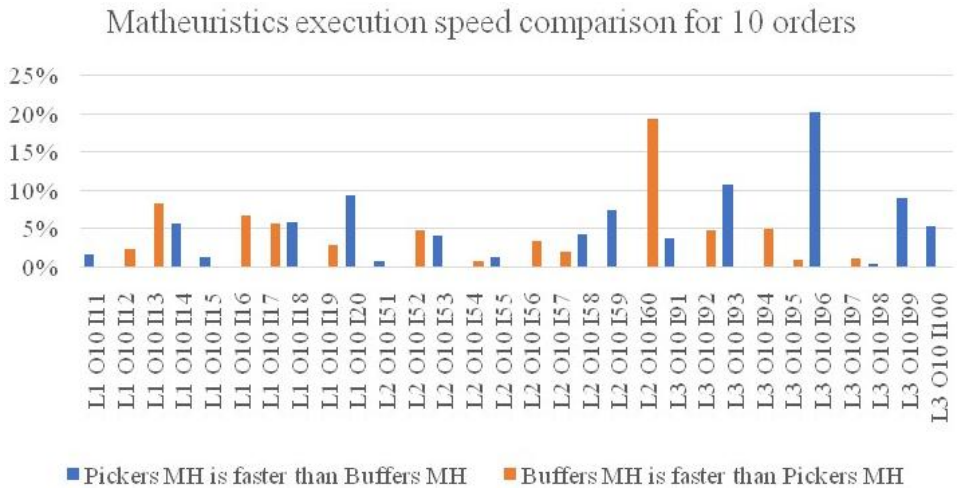

Fig. 13. 2-step Pickers, Buffers matheuristics execution speed comparison for 10 orders: L{1,2,3} – layouts, O10–10 orders, I {11..20, 51..60, 91–100} – instances

time for Buffers matheuristics was, on average, 1.51 seconds. For the 3rd layout, the solution time for Pickers matheuristics was, on average, 2.32 seconds. For the 3rd layout, the solution time for Buffers matheuristics was, on average, 2.43 seconds.

Figure 14 presents the 2-step matheuristics execution speed comparison for 15 orders. For the 1st layout L1, Pickers matheuristics were faster than Buffers matheuristics in 6 instances on average at 9.15%, while Buffers matheuristics were faster than Pickers matheuristics in 4 instances on average at 8.90%. For the 2nd layout L2, Pickers matheuristics were faster than Buffers matheuristics in 2 instances on average at 6.54%, while Buffers matheuristics were faster than Pickers matheuristics in 8 instances on average at 23.69%. For the 3rd layout L3, Pickers matheuristics were faster than Buffers matheuristics in 6 instances on average at 10.04%, while Buffers matheuristics were faster than Pickers matheuristics in 3 instances on average at 2.20%.

For the 1st layout, the solution time for Pickers matheuristics was, on average, 3.51 seconds. For the 1st layout, the solution time for Buffers matheuristics was, on average, 3.58 seconds. For the 2nd layout, the solution time for Pickers matheuristics was, on average, 30.16 seconds. For the 2nd layout, the solution time

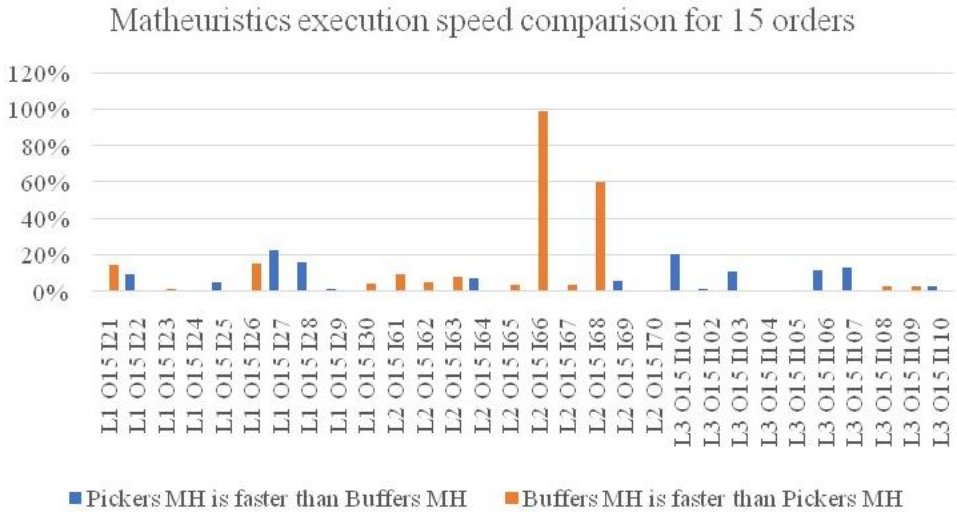

Fig. 14. 2-step Pickers, Buffers matheuristics execution speed comparison for 15 orders: L{1,2,3} – layouts, O15–15 orders, I {21..30, 61..70, 101–110} – instances

for Buffers matheuristics was, on average, 3.09 seconds. For the 3rd layout, the solution time for Pickers matheuristics was, on average, 4.21 seconds. For the 3rd layout, the solution time for Buffers matheuristics was, on average, 4.46 seconds.

Figure 15 presents the 2-step matheuristics execution speed comparison for 20 orders. For the 1st layout L1, Pickers matheuristics were faster than Buffers matheuristics in 5 instances on average at 8.76%, while Buffers matheuristics were faster than Pickers matheuristics in 5 instances on average at 22.27%. For the 2nd layout L2, Pickers matheuristics were faster than Buffers matheuristics in 2 instances on average at 34.93%, while Buffers matheuristics were faster than Pickers matheuristics in 8 instances on average at 7.25%. For the 3rd layout L3, Pickers matheuristics were faster than Buffers matheuristics in 6 instances on average at 11.26%, while Buffers matheuristics were faster than Pickers matheuristics in 4 instances on average at 11.67%.

For the 1st layout, the solution time for Pickers matheuristics was, on average, 7.71 seconds. For the 1st layout, the solution time for Buffers matheuristics was, on average, 5.18 seconds. For the 2nd layout, the solution time for Pickers matheuristics was, on average, 4.42 seconds. For the 2nd layout, the solution time for Buffers matheuristics was, on average, 4.45 seconds. For the 3rd layout, the solution time for Pickers matheuristics was, on average, 6.65 seconds. For the 3rd layout, the solution time for Buffers matheuristics was, on average, 6.79 seconds.

Figure 16 shows the distance to the lower bound (LB) of the Full model solution for 5 orders. For the 1st layout L1, an optimal solution for all 10 instances was found. For the 2nd layout L2, an optimal solution only for 1 instance was found. The other 9 instances were far from LB on average at 33.88%, with minimal and maximal values at 11.81% and 59.14%. For the 3rd layout L3, an optimal solution for 8 instances was found. The other 2 instances were far from LB on average at 11.19%.

Figure 17 shows the distance to LB of the Full model solution for 10 orders. For the 1st layout L1, an optimal solution for 6 instances was found. The remaining 4 instances were far from LB on average at 15.07%. For the 2nd layout L2, an optimal solution only for 1 instance was found. The other 9 instances were far from LB on average at 49.43%, with minimal and maximal values of 29.46% and 60.10%. For the 3rd layout L3, an optimal solution for 8 instances was found. The other 2 instances were far from LB on average at 9.08%.

Figure 18 shows the distance to LB of the Full model solution for 15 orders. For the 1st layout L1, an optimal solution was not found for any instances. All instances were far from LB on average at 50.69%, with minimal and maximal values of 35.96% and 61.79%. For the 2nd layout L2, an optimal solution was not found for any instances. All instances were far from LB on average at 71.31%, with minimal and maximal values at 54.92% and 78.73%. For the 3rd layout L3, an optimal solution was not found for any instances. All instances were

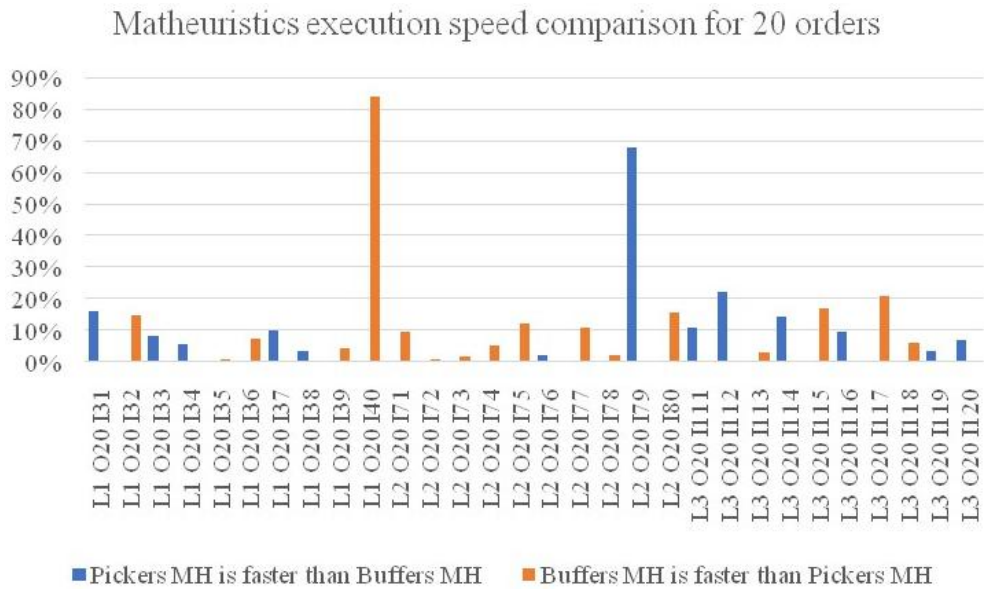

Fig. 15. 2-step Pickers, Buffers matheuristics execution speed comparison for 20 orders: L{1,2,3} – layouts, O20–20 orders, I {31..40, 71..80, 111–120} – instances

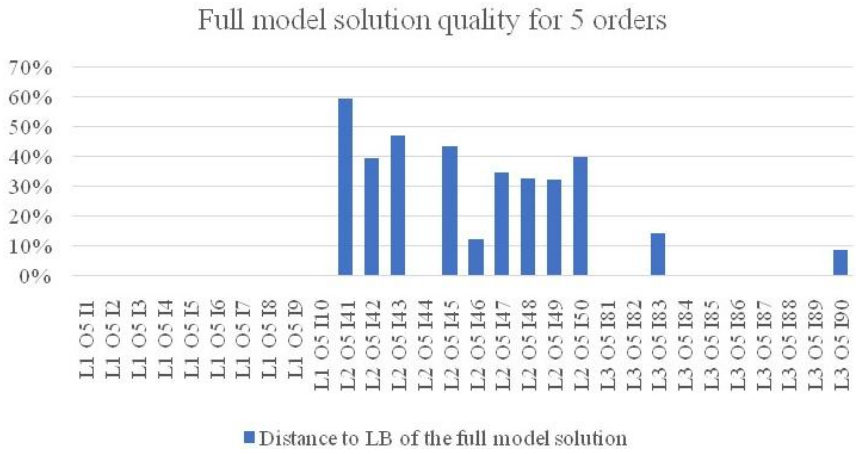

Fig. 16. Distance to LB of the Full model solution for 5 orders: L{1,2,3} – layouts, O5–5 orders, I {1..10, 41..50, 81–90} – instances

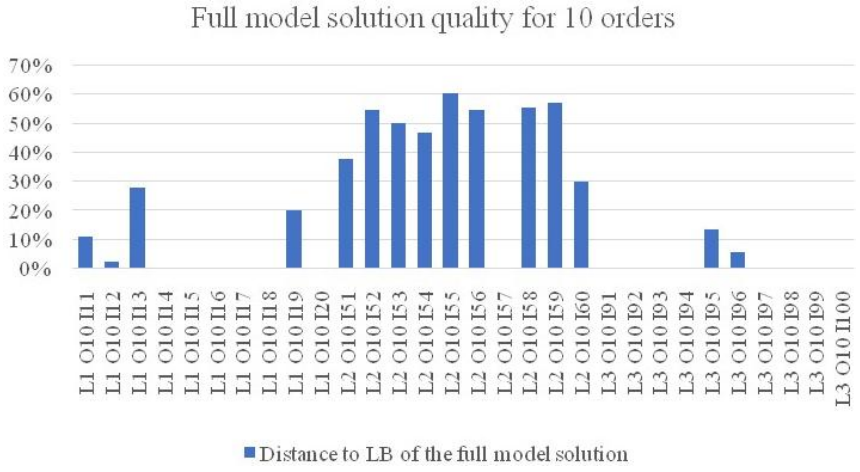

Fig. 17. Distance to LB of the Full model solution for 10 orders: L{1,2,3} – layouts, O10–10 orders, I {11..20, 51..60, 91–100} – instances

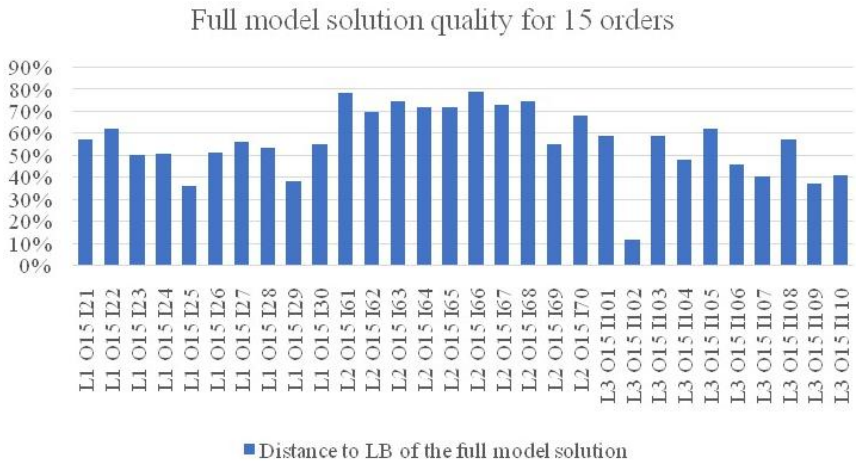

Fig. 18. Distance to LB of the Full model solution for 15 orders: L{1,2,3} – layouts, O15–15 orders, I {21..30, 61..70, 101–110} – instances

far from LB on average at 45.77%, with minimal and maximal values at 11.16% and 61.54%.

Figure 19 shows the distance to LB of the Full model solution for 20 orders. For the 1st layout L1, an optimal solution was not found for any instances. All instances were far from LB on average at 59.47%, with minimal and maximal values of 52.99% and 69.72%. For the 2nd layout L2, an optimal solution was not found for any instances. All instances were far from LB on average at 74.93%, with minimal and maximal values of 70.21% and 81.27%. For the 3rd layout L3, an optimal solution was not found for any instances. All instances were far from LB on average at 63.13%, with minimal and maximal values of 52.69% and 70.24%.

Computational experiments performed by the authors have shown the performance of the developed 2-step Pickers, Buffers matheuristics. Success or failure may depend on one's ability to assess and control algorithmic efficiency. It is preferable to use as few resources as possible for optimal effectiveness.

Pickers and Buffers matheuristics found the same solutions for 80 of 120 instances. Pickers matheuristics were better than Buffers matheuristics in 18 of 120 instances (or of 40 instances for which the solution differed) on average at 15.61%. Buffers matheuristics were better than Pickers matheuristics in 22 of 120 instances (or of 40 instances for which the solution differed) on average at 13.98%.

Compared to the Full model solution, matheuristics were better in 85 of 120 instances. Pickers matheuristics were better than the Full model solution in 85 of 120 instances on average at 46.56%. Buffers matheuristics were better than the Full model solution in 85 of 120 instances on average at 46.87%.

Compared to both matheuristics the Full model solution was better in 19 of 120 instances. The Full model solution was better than Pickers matheuristics in 18 instances on average at 13.59%. The Full model solution was better than Buffers matheuristics in 18 instances on average at 14.11%.

Pickers matheuristics were faster than Buffers matheuristics in 57 instances of 120 instances on average at 8.87%, while Buffers matheuristics were faster than Pickers matheuristics in 60 instances on average at 9.48%. The solution time for Pickers matheuristics was, on average, 5.41 seconds and varied from 0.57 seconds to 4.47 minutes. The solution time for Buffers matheuristics was, on average, 2.99 seconds and varied from 0.56 seconds to 8.17 seconds.

The optimal solution was found by the Full model solution for 34 of 120 instances. The remaining 86 instances were far from LB on average at 52.76%, with minimal and maximal values at 2.08% and 81.27%, respectively.

Conclusion

In this research, we model the logistics order-picking problem with sequence-dependant constraints and a one-directional conveyor with two decision variables: container start time and product quantity. We propose the two variants of the 2-step matheuristics in

Fig. 19. Distance to LB of the Full model solution for 20 orders: L{1,2,3} – layouts, O20–20 orders, I {31..40, 71..80, 111–120} – instances

which the first step is the Pickers or Buffers free time minimization problem and finding the value of the product quantity; the second step is order picking makespan minimization problem and finding the container start time using the product quantity achieved on the first step. Both steps were solved using mathematical programming. In the second step, we included several simplification approaches, which significantly reduced the solution time, they are: excluding not-ordered products, excluding not-in-buffers products, excluding products without quantities found in the first step of matheuristics.

Heuristics and matheuristics are effective functions that can be used consciously or unconsciously and ignore some information. We use this method to split the main problem with two decision variables into two steps, where in each step, we are looking for one decision variable. We delivered a workable method for solving the order picking problem that anyone can implement in the logistics center. The developed matheuristics result in good decision-making for order-picking scenarios in various DC layouts.

Matheuristics are versatile and can be tailored to the specific characteristics of optimization problems. The combination of mathematical programming and heuristic methods provides a powerful framework for solving complex problems in various domains.

In this study, we performed computational experiments for 3 versions of the DC layouts, which differ with pickers to buffers assignment: a picker completes orders in a single buffer, a picker completes orders in several buffers, and several pickers complete orders in one buffer. The results of the matheuristics were compared between themselves and the mathematical programming of the Full model solution. For 80 of 120 instances, Pickers and Buffers matheuristics found identical results. On the one hand, Pickers matheuristics were better than Buffers matheuristics in 18 of 120 instances (or of 40 instances with unique results) on average at 15.61%. On the other hand, Buffers matheuristics were better than Pickers matheuristics in 22 of 120 instances (or of 40 instances with unique results) on average at 13.98%. Compared to the Full model solution, Pickers and Buffers matheuristics were better in 85 of 120 instances. Pickers matheuristics were better than the Full model solution on average at 46.56%, while Buffers matheuristics were better on average at 46.87%.

The computational experiment results suggest several key findings regarding the performance of matheuristics compared to the Full model for different scenarios.

- The results indicate that as the number of orders increases, the quality of the Full model solution decreases. Specifically, for 15 and 20 orders, matheuristics consistently outperformed the Full model across all instances.
- This highlights the scalability and efficiency of matheuristics in handling larger problem instances. The ability of matheuristics to provide better solutions with an increasing number of orders makes them a favorable choice for real-world scenarios with dynamic demand.
- The 2nd layout L2 was identified as the most complicated for the Full model solution on small instances. Matheuristics consistently delivered higher-quality solutions for this layout in such cases. This highlights the fact that matheuristics are very useful for handling complex layouts. Compared to conventional Full models, they are better able to handle complex configurations because of their flexibility and adaptability.
- The findings from the study support the proposal to use matheuristics, particularly for cases with increased order numbers and complex layouts.
- Because heuristics consistently found better solutions across all layouts, they are recommended for both small and large cases. This wide range of applications highlights matheuristics' adaptability in solving various optimization problems.
- Knowledge of the matheuristics' possibilities can help businesses to stay competitive.

Although there are advantages and disadvantages to utilizing heuristics and matheuristics in decision-making, they are not naturally positive or negative. They might speed up the process of issue-solving, but they can also result in incorrect evaluations of third parties or circumstances.

Matheuristics are effective in this way, which is a key factor in why they should be used in the DC. As there is no certainty of their veracity, the distributor should be cautious about how much he relies on them.

The developed matheuristics enable DC planners to schedule orders without much effort while having differing justifications. They obtained solutions fast and didn't take up much of the available resources.

This work provides a foundation for a variety of future research opportunities. For instance, it is useful to allow the storage of products in one location, as in this research, some kinds of products were stored in several locations in different buffers. This complicated the problem because the container must go to the next round if the next product is located in the previous buffer. In this research, because the products were stored in several buffers, the container could travel to the second location where the product is stored, especially if it is on the way of the conveyor moving direction.

Moreover, we propose to add differentiation based on the product type. In this research, the product could be picked only from one location (buffer) because the distributor cannot guarantee the identity of, for example, colors in different deliveries. So we propose to add into the model the case if the product is totally identical in different deliveries and some items of it could be taken from one buffer and the rest parts could be taken from another buffer and placed into the same order.

In future research, the proposed matheuristics could be utilized by artificial intelligence and machine learning programs to guide their output.

Acknowledgments

The project is co-financed by the European Union under the European Regional Development Fund program under the Intelligent Development Program. The project is carried out as part of the National Center for Research and Development's Program "Szybka ścieżka" (project no. POIR.01.01.01-00-0352/22). The studies were realized with support from statutory activity financed by the Polish Ministry of Science and Higher Education (0613/SBAD/4821).

References

Beauchemin, M., Ménard, M.-A., Gaudreault, J., Lehoux, N., Agnard, S., & Quimper, C.-G. (2022). Dynamic allocation of human resources: case study in the metal 4.0 manufacturing industry. *International Journal of Production Research*, 1–17. DOI: 10.1080/00207543.2022.2139002.

Chiang, D.M.H., Lin, C.P., & Chen, M.C. (2011). The adaptive approach for storage assignment by mining data of warehouse management system for distribution centres. *Enterprise Information Systems*, *5*(2), 219–234. DOI: 10.1080/17517575.2010.537784.

Cinar, D., Oliveira, J.A., Ilker Topcu, Y., & Pardalos, P.M. (2017). Scheduling the truckload operations in automated warehouses with alternative aisles for pallets. *Applied Soft Computing*, *52*, 566–574. DOI: 10.1016/J.ASOC.2016.10.013.

Czerniachowska, K., Wichniarek, R., & Żywicki, K. (2023a). A Model for an Order-Picking Problem with a One-Directional Conveyor and Buffer. *Sustainability*, *15*(18), 13731. DOI: 10.3390/su151813731.

Czerniachowska, K., Wichniarek, R., & Żywicki, K. (2023b). A Two-Step Matheuristics for Order-Picking Process Problems with One-Directional Material Flow and Buffers. *Applied Sciences*, *13*(18), 10099. DOI: 10.3390/app131810099.

Czerniachowska, K., Wichniarek, R., & Żywicki, K. (2023c). Constraint Programming for Flexible Flow Shop Scheduling Problem with Repeated Jobs and Repeated Operations. *Advances in Science and Technology Research Journal*, 17(3), 280–293. DOI: 10.12913/22998624/166588.

Da Col, G., & Teppan, E.C. (2019). Google vs IBM: A constraint solving challenge on the job-shop scheduling problem. *Electronic Proceedings in Theoretical Computer Science, EPTCS*, *306*, 259–265. DOI: 10.4204/EPTCS.306.30.

Da Col, G., & Teppan, E.C. (2022). Industrial-size job shop scheduling with constraint programming. *Operations Research Perspectives*, *9*, 100249. DOI: 10.1016/j.orp.2022.100249.

Danilczuk, W., Gola, A., & Grznar, P. (2022). Job Scheduling Algorithm for a Hybrid MTO-MTS Production Process. *IFAC-PapersOnLine*, *55*(2). DOI: 10.1016/j.ifacol.2022.04.235.

De Koster, R., Le-Duc, T., & Roodbergen, K.J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, *182*(2), 481–501. DOI: 10.1016/j.ejor.2006.07.009.

Fuchigami, H.Y., & Rangel, S. (2018). A survey of case studies in production scheduling: Analysis and perspectives. *Journal of Computational Science*, *25*, 425–436. DOI: 10.1016/j.jocs.2017.06.004.

Gu, J., Goetschalckx, M., & McGinnis, L.F. (2007). Research on warehouse operation: A comprehensive review. *European Journal of Operational Research*, *177*(1), 1–21. DOI: 10.1016/j.ejor.2006.02.025.

Haouassi, M., Kergosien, Y., Mendoza, J.E., & Rousseau, L.M. (2022). The integrated orderline batching, batch scheduling, and picker routing problem with multiple pickers: the benefits of splitting customer orders. *Flexible Services and Manufacturing Journal*, *34*(3), 614–645. DOI: 10.1007/s10696-021-09425-8.

Hsu, S.Y., & Liu, C.H. (2009). Improving the delivery efficiency of the customer order scheduling problem in a job shop. *Computers & Industrial Engineering*, *57*(3), 856–866. DOI: 10.1016/J.CIE.2009.02.015.

Hultkrantz, O., & Lumsden, K. (2001). E-commerce and consequences for the logistics industry. *Oecd the Impact of E-Commerce on Transport*, 1–15.

Iwasaki, Y., Suzuki, I., Yamamoto, M., & Furukawa, M. (2013). Job-shop Scheduling Approach to Order-picking Problem. *Transactions of the Institute of Systems, Control and Information Engineers*, *26*(3), 103–109. DOI: 10.5687/iscie.26.103.

Kawęcki, N., & Gola, A. (2022). Pick Performance System as an IT Support for Order Completing – A Case Study. *Lecture Notes in Mechanical Engineering*, 105–115. DOI: 10.1007/978-3-030-99310-8_9.

Kuthambalayan, T.S., & Bera, S. (2020). Managing product variety with mixed make-to-stock/make-to-order production strategy and guaranteed delivery time under stochastic demand. *Computers and Industrial Engineering*, *147*, 106603. DOI: 10.1016/j.cie.2020.106603.

Lee, H.Y., & Murray, C.C. (2019). Robotics in order picking: evaluating warehouse layouts for pick, place, and transport vehicle routing systems. *International Journal of Production Research*, *57*(18), 5821–5841. DOI: 10.1080/00207543.2018.1552031.

Liu, C.M. (1999). Clustering techniques for stock location and order-picking in a distribution center. *Computers and Operations Research*, *26*(10–11), 989–1002. DOI: 10.1016/S0305-0548(99)00026-X.

Nowicki, E., & Smutnicki, C. (1996). A fast taboo search algorithm for the job shop problem. *Management Science*, *42*(6), 797–813. DOI: 10.1287/mnsc.42.6.797.

Onal, S., Zhu, W., & Das, S. (2023). Order picking heuristics for online order fulfillment warehouses with explosive storage. *International Journal of Production Economics*, *256*, 108747. DOI: 10.1016/j.ijpe.2022.108747.

Peeters, K., & van Ooijen, H. (2020). Hybrid make-to-stock and make-to-order systems: a taxonomic review. In *International Journal of Production Research* (Vol. 58, Issue 15, pp. 4659–4688). DOI: 10.1080/00207543.2020.1778204.

Petersen, C.G. (2000). An evaluation of order picking policies for mail order companies. *Production and Operations Management*, *9*(4), 319–335. DOI: 10.1111/j.1937-5956.2000.tb00461.x.

Pinto, A.R.F., Nagano, M.S., & Boz, E. (2023). A classification approach to order picking systems and policies: Integrating automation and optimization for future research. *Results in Control and Optimization*, *12*, 100281. DOI: 10.1016/j.rico.2023.100281.

Roodbergen, K.J., & De Koster, R. (2001). Routing methods for warehouses with multiple cross aisles. *International Journal of Production Research*, *39*(9), 1865–1883. DOI: 10.1080/00207540110028128.

Roodbergen, K.J., Vis, I.F.A., & Taylor, G.D. (2015). Simultaneous determination of warehouse layout and control policies. *International Journal of Production Research*, *53*(11), 3306–3326. DOI: 10.1080/00207543.2014.978029.

Roundy, R., Chen, D., Chen, P., Çakanyildirim, M., Freimer, M.B., & Melkonian, V. (2005). Capacity-driven acceptance of customer orders for a multi-stage batch manufacturing system: Models and algorithms. *IIE Transactions (Institute of Industrial Engineers)*, *37*(12), 1093–1105. DOI: 10.1080/07408170500288042.

Sadeh, N., & Fox, M.S. (1996). Variable and value ordering heuristics for the job shop scheduling constraint satisfaction problem. *Artificial Intelligence*, *86*(1), 1–41. DOI: 10.1016/0004-3702(95)00098-4.

Tang, L.C., & Chew, E.P. (1997). Order picking systems : Batching and storage assignment strategies. *Computers and Industrial Engineering*, *33*(3–4), 817–820. DOI: 10.1016/s0360-8352(97)00245-3.

van Gils, T., Ramaekers, K., Caris, A., & de Koster, R.B.M. (2018). Designing efficient order picking systems by combining planning problems: State-of-the-art classification and review. *European Journal of Operational Research*. DOI: 10.1016/j.ejor.2017. 09.002.

Wu, Z.H., Chen, H.J., & Yang, J.J. (2020). Optimization of Order-Picking Problems by Intelligent Optimization Algorithm. *Mathematical Problems in Engineering*, *2020*. DOI: 10.1155/2020/6352539.