SZÉCHENYI ISTVÁN UNIVERSITY

Theses of Doctoral Dissertation

Tamás Bódis

Széchenyi István University
Audi Hungaria Faculty of Automotive Engineering
2019

Tamás Bódis

Development of order picking algorithms based on product stacking factors

Theses of Doctoral Dissertation

Supervisor:
János Botzheim, Ph.D.
Associate Professor
Budapest University of Technology and Economics

Multidisciplinary Doctoral School of Engineering Sciences
Széchenyi István University

Contents

List of Figures

List of Tables

List of Acronyms

Acronym	**Meaning**
ABC	Artificial Bee Colony
ACO	Ant Colony Optimisation
BM	Bacterial Mutation
BMA	Bacterial Memetic Algorithm
CLP	Container Loading Problem
DA	Departure and arrival (position)
DE	Differential Evolution
GA	Genetic Algorithm
GT	Gene Transfer
HQ	High Quantity
KPI	Key Performance Indicator
LQ	Low Quantity
LS	Local Search
OPRP-PLF	Order Picking Routing Problem based on Pallet Loading Feature
PC	Product Class
PLC	Pallet Loading Class
PLF	Pallet Loading Feature
PLFDM	Pallet Loading Feature based Decision Matrix
PLP	Pallet Loading Problem
PLR	Pallet Loading Rate
POPC	Product and Order Parameter based Class
PSO	Particle Swarm optimisation
RND	Randomised
SA	Simulated Annealing
SKU	Stock Keeping Unit
SLA	Storage Location Assignment
SPOPC	Special Product and Order Parameter based Class
TSP	Travelling Salesman Problem
UL	Unit Load
VRP	Vehicle Routing Problem
WMS	Warehouse Management Systems

Chapter 1

Introduction

Warehouses are integrated parts of the global supply chains. Their general function is to support the material supply, the production and the distribution processes from the raw material productions and the work-in-progress states through to the finished goods. They handle and store products in the storage system and prepare the ordered units for transport. The current warehousing challenges are the increasing product range, the product and order customisation, the fluctuating demands, the decreasing ordered quantities, and the shortening of the available time for warehousing operations. Because of these challenges, the warehouses need to be adaptive and should continuously develop the warehousing system and the processes to increase and maintain the customer services level. The high human resource, equipment and infrastructure requirements make the warehouses a costly part of the supply chains, which increases the necessity of well-designed, well-organised, and continuously developed warehousing systems [58, 42].

While order picking is the most costly and labour intensive warehousing operation, most of the research works and industrial projects are focusing on this field. During the order picking the pickers collect the ordered products based on the order picking list and build unit loads to satisfy customer demands. The order picking list generally contains numerous ordered products with ordered quantity and usually defines the customer specific unit load building methodology. Each ordered product has at least one order picking position, which is visited by the picker to pick the right amount of Stock Keeping Units onto a pallet and build a stable unit load.

The unit load (UL) merges the ordered Stock Keeping Units (SKU) into a single unit, which can be easily and safely handled by any material handling equipment (e.g., pallet jack, reach truck, etc.). SKU is a distinct and inseparable unit of the products [61]. The ULs are compatible with storage-, material handling-, and transport systems because of the applied standardised pallets and packaging solutions. Packaging solution of a SKU can be for example bag, bin, box, can or tray, which is responsible for containing and protecting the product, and supporting the material handling [1, 70].

The effectiveness of the pickers depends on the synchronisation of several processes, decisions, and factors.

The typical decision problems in design and control of order picking processes are the routing methods, layout design, Storage Location Assignment (SLA) methods, order separation, order batching, and zoning. While travelling time is approximately 50% of the whole picking time, the primary goal of the order picking process development is the routing optimisation. It is responsible for sequencing the picking position of the products on the order picking list in order to get the shortest route length through the warehouse. It can be interpreted as a special case of the classical Travelling Salesman Problem (TSP) or Steiner TSP [42]. The routing is strongly influenced by the storage location assignment and the layout design. The storage location assignment methods allocate products to picking positions, which are visited by the pickers. The layout design determines the number and the dimensions of the storage blocks, the position of the departure and arrival point of the order picking on the manipulation area, and the number, length, and width of the aisles. These factors have a huge impact on the picking sequence of ordered products, routing distance, order picking lead time, and material handling cost [42, 21]. When the ordered products can not be picked onto one pallet, then it is necessary to separate the customer order to order picking lists for each required UL. Furthermore, when the ordered quantity of one order does not utilise the UL, then batching orders for one picking unit can shorten the picking lead time. However, after the picking the products should be sorted for customers. In both cases the order picking list generating method should define the contents of the picking list for the purpose of minimising the summarised order fulfilment lead time. Separated order picking zones have also an impact on the order picking list definition. When sequential or parallel zone picking is applied, then the order picking list should be separated for zones.

Besides the generally discussed order picking related research fields, the Container Loading Problem has also connections to the order picking. Its algorithms are responsible for assigning three-dimensional small products to three-dimensional rectangular large objects (i.e., truck, containers, pallet). Its aim is to hold the basic geometric feasibility conditions and reach the defined problem specific objective function [13]. Bortfeldt and Wäscher collected and structured the main objective functions and problem types of the Container Loading Problem, where Bin Packing is a minimisation problem. Generally it is responsible for assigning strongly heterogeneous products into a minimum number of containers. In the case of warehouses, Bin Packing algorithms are used, for example, when the customer order has to be separated to ULs, because of, for example, a large quantity order or a high number of products [13]. The Pallet Loading Problem (PLP) is a maximisation problem, which is responsible for packing the maximum number of identical rectangular boxes onto a rectangular pallet [2]. In the case of warehouses the PLP answers the question of how to position the products on the pallet. These products can be defined to this pallet by a Bin Packing algorithm. The different characteristics of orders, the various attributes of the packages, and the UL making requirements of different partners make the Bin Packing and Pallet Loading Problem complex.

Each influencing parameter of order picking has an impact on the others with a different importance, which highlights the complexity of warehousing processes.

Many researchers work on the different segments of the order picking, Bin Packing or PLP development. Several researchers have summarised, evaluated, and developed TSP solutions for order picking operations (e.g., [67, 20, 62]). Most of the solutions assume that the physical parameters of products allow any stacking sequence on the pallet. The routing and loading aspects have been taken into consideration in Vehicle Routing Problem (VRP) algorithms in the field of transport logistics, whose sub-problems have been summarised by Pollaris et al.. The order picking routing problem could be a special case of the VRP with loading constraints when one picker has one set of positions. The positions have to be visited and the products should be arranged on a pallet to build a stable UL [56, 18].

Many solutions have been defined for harmonising SLA and routing to decrease the routing distances and times (e.g., [48, 19]). However, while the physical product parameters (dimensions, weight, packaging), the product stacking attributes, and the order characteristics influence the physically possible picking sequence in order to build stable ULs, researchers rarely take into account these aspects during SLA and order picking routing optimisation. From another perspective, many researchers have attained valuable results in the fields of Pallet Loading and Bin Packing Problem (e.g., [46, 43, 5, 60, 26]), but the solutions are rarely harmonised with SLA and order picking routing algorithms.

Molnár and Lipovszki highlighted the importance of a well sequenced order picking list to support well structured and stable ULs to avoid product damages [49]. While Molnár and Lipovszki developed routing optimisation by considering product stacking attributes, they determined picking sequence of product classes. Their algorithm minimises the difference from the defined sequence and minimises the distance but sometimes a more flexible and a more complex sequencing rule definition could be required, which depends not only on the product parameters [49].

The following simple everyday example would like to introduce the proposed research problem. We are going to a supermarket with our shopping list which contains the following products: 2 kg potatoes, 2 dozen eggs, 2 bottle soft drinks, 2 chips, and 2 marzipan figures. We pick all products into our shopping basket. Our goal is to pick products with the shortest lead time, the shortest distance, and without product damages. If we find the marzipan figures, the chips, and the eggs near the entrance, and later we put the potatoes and the soft drinks on the bottom of the already picked products, most of the products will be damaged before paying. Naturally, usually we pick the heavy and the less sensitive products first (soft drinks, potato) or reconstruct the product sequence within the shopping basket during "picking". If the product location assignment supports the right, risk free picking sequence, we will pick our demands on the shortest way without reconstructions. Otherwise we have to decide, whether we collect products on the right sequence and walk more or pick them with the shortest routing and redesign the contents of the basket during the picking. Both solutions can result in the shortest picking lead time. The best choice depends on the length and the content of our list, and on the product allocation in the supermarket.

The above mentioned shopping example is simple, but it is well related to the order picking process. The shopping list is the order picking list, whose characteristic has an impact on the physical processes (picking sequence, routing, UL reconstruction, etc.) The shop entrance is the departure position where the order picking is started with basket pick-up. The basket is the UL making equipment, which is usually the pallet in a warehouse. The shelves are the order picking positions where the products are allocated. The SLA is also relevant in the shops, but it focuses more on marketing aspects and less on picking effectiveness. The cashier is the arrival position of the warehouse shipping area, where the final control, the administrations, and the truck loading happen.

The proposed example highlights the impacts of the order characteristics, the departure and arrival position, UL reconstruction during picking, the routing and the SLA on the sequencing decision of the picker. The general goal of the picker is to minimise the order picking lead time and build stable ULs without product damages. The order picking system design should synchronise each described decision fields and consider each necessary aspects (e.g, product stacking possibilities) to support the pickers in an effective order picking of different characteristics orders [10].

1.1 Research goals and motivation

I realised during industrial projects as a logistics consultant, that the stacking attribute of the packages, and the UL building possibilities and rules could have a huge impact on the effectiveness of the order picking. Where these aspects are relevant and exact algorithms are not available, the pickers have huge challenges to manage the order picking process. They should take into consideration several factors using their brain to find the shortest picking lead time, to build stable transport units, and to avoid product damages. These challenges are usually handled by best practices in the industry. However, we can realise based on industrial experiences, that synchronisation of the order picking routing, the storage location assignment, and the product stacking attribute based unit load making have a huge impact on the order picking lead time and the operational cost, these aspect have not been discussed and harmonised comprehensively by the order picking research works yet.

Based on my industrial experiences and state of the art research I set my following goals to develop industrially relevant and scientifically unique order picking solutions based on product stacking factors.

- I highlight and define the Order Picking Routing Problem based on Pallet Loading Feature (OPRP-PLF) as a novel and complex problem, and prove its necessity. I define the Pallet Loading Features (PLF), which depends on product attributes, order picking list characteristics, and order picking system.
- I build a formalised, flexible, parametric, and industrially relevant model for the OPRP-PLF. This model should be defined based on known, easily measurable and rarely changing data, because proper product parameters (geometric, weight) are rarely available.
- I develop a methodology for examining, when it is necessary to implement an OPRP-PLF algorithm at a warehouse.
- I examine the complexity of my problem to find the right optimisation methodology.
- I develop, evaluate, and compare algorithms for the OPRP-PLF, which can support the pickers with time effective picking sequence within the available time. The solution should ensure flexibility, avoid product damages, support the stable unit load building, and minimise the order picking lead time.
- I examine the effects of allowing unit load reconstruction during order picking on the order picking lead time.
- I examine the effects of the warehouse layout, the PLF based SLA and routing synchronisation on the order picking effectiveness. I would highlight, that applying PLF aspects during SLA has an impact on the order picking lead time. Defining the right PLF based SLA algorithm is a possible further research, but not the scope of my proposed research.

Besides my research, I would like to apply my state of the art research and my scientific results into education as new challenges, aspects, and solutions of order picking.

1.2 Dissertation structure and methodology

My proposed research has two main chapters. Chapter 2 summarises my literature review of the state of the art research in the field of order picking and optimisation. The main points of the order picking state of the art research (Section 2.1) are the routing, the storage location assignment, the layout, the zoning and the order picking list definition, which are connecting to my research. Section 2.2 summarises the relevant optimisation terminologies and methodologies. Chapter 3 introduces my own results based the following structure.

First of all Section 3.1 defines the OPRP-PLF, explains its influencing factors and highlights the relevance of the industrial applications. I describe my model for formulating the pallet loading rules by mathematical formulas. This section introduces my methodologies to define the necessity of applying OPRP-PLF algorithms at a warehouse. My solutions can highlight when OPRP-PLF algorithms should be implemented for a warehouse based on the analysis of the pallet loading rules and monitoring the nature of the warehousing processes.

Section 3.2 examines the complexity of industrially relevant sub-problems of the OPRP-PLF based on a mathematical methodology. I determine formula for each case to calculate the possible number of order picking sequencing variations and I examine the behaviour of those in the case of order picking lists whose length and contents are different. The aim of the complexity evaluation is to find the necessary optimisation methodology for the problem.

I explain the details of my own developed algorithms for OPRP-PLF in Section 3.3. I describe my objective function and highlight the necessity of optimisation based on analytic examination of simple examples. I introduce alternative solutions for Bacterial Memetic Algorithm operators and Simulated Annealing algorithms. Pseudo codes and graphics make the solutions understandable and reproducible. The combinations of the mentioned operators will define possible algorithm solutions, which will be evaluated and compared in following section.

Section 3.4 introduces the evaluation of my previously defined possible algorithm alternatives. I examine the algorithms by my own developed computer simulation environment, which is developed for the proposed problem. I evaluate the alternatives with my objective function based on order picking list whose length and contents are different. The aim of this section is to find the relevant algorithms operators and define the algorithm(s) for further application.

Section 3.5 examines the effects of the warehouse layout attributes, the SLA and the UL reconstruction on the order picking lead time. I determine industrially relevant alternatives for these system attributes to evaluate them on several orders with different characteristics by the simulation environment with applying the defined OPRP-PLF algorithm(s). Based on the objective evaluation of alternative system configuration I will make consequences on warehousing logistics point of view.

Section 4 summarises my research results and collects my further research.

Chapter 2

State of the art

2.1 Order Picking State of the Art

The aim of this section is to overview the warehousing logistics terminology and the state of the art results, which are related to the proposed research. Koster et al. made a well-structured overview of order picking, which will be the basis of my state of the art research [42]. It will be completed with uptodate results and the relating research areas will be discussed. I will summarise the considered aspects and the conclusions of the examined papers.

Warehouses could have several functions, like raw material or finished good warehouse of factories, cross-docking warehouses or distributional centres. This research focuses on warehouses, where thousands of products are handled and order picking is a major operation.

Koster classified the order picking systems (Figure 1.), where the first classes are based on employment of the human resources or machineries. When pickers perform the customer orders there are possibilities for different levels of automation. In the case of picker-to-part warehouses the picker moves to the order picking positions, where products are stored. The parts-to-picker solutions require automation, because the pickers stay at the picking position and the products are moved there for picking and after the picking the UL is stored back again. The put systems has two steps. The first one is the preparation, when the ordered products are pre-picked for order picking. The second one is, when the prepared products are separated for customer orders. The picker-to-parts order picking systems are separable into low level and high level systems. While the high level systems require any lifting solutions during picking, in the case of low level order picking systems the picker is able to reach the picking positions from the floor. The picker-to-parts systems allow many process management variants. One of the possibility is the way of defining the order picking list. In the case of pick by article, the orders are batched, the summarised quantity of products are picked and separated for the orders.

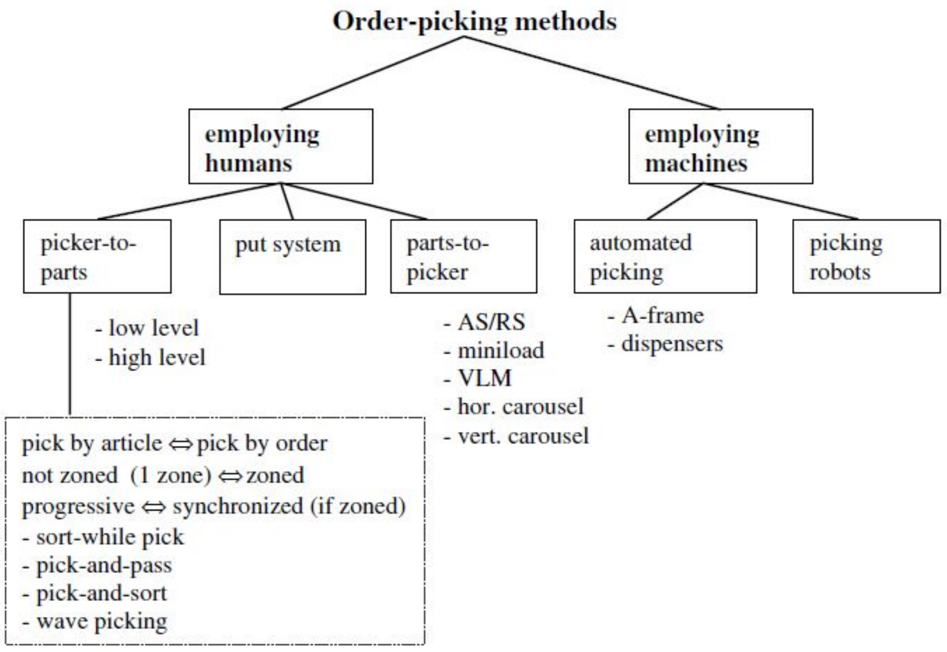


Figure 1: Classification of order-picking systems [41, 42]

The discrete picking methodology allows to pick the ordered products of a customer or a part of it to define one UL. The UL can be defined during (sort-while-pick) and after the picking (pick-and-sort). The picking lists and the tasks can be separated for zone, when the picking processes can be sequential (progressive) and parallel (synchronised). In the case of sequential picking the picker collects the products in his/her dedicated zone, then forward the picked UL and the picking list to the next zone, where the next picker will continue the picking (pick-and-pass). The parallel picking allows to start the picking of a customer order in each separated zone at the same time, but the zone by zone made ULs should be merged and consolidated after the picking [41, 42].

The proposed research concentrates on low level and not zoned picker-to-parts order picking system, where discrete picking and sort-while-pick methodologies are applied.

Koster et al. defined the main decision fields of the order picking system design, which gives the structure of the following state of the art research [42].

- Routing
- Storage Location Assignment (SLA)
- Layout
- Zoning
- Batching - Defining the order picking lists

2.1.1 Routing

According to Tompkins et al., travelling is the most time-consuming movement of manual, picker-to-parts order picking processes [68]. Tompkins et al. highlighted the distribution of the order picker's time between the typical order picking movements [68]. Figure 2 illustrates, that the "Travel" gives approximately 50% of the order picking time, when the picker is moving between positions. Further movement is the "Search", when the picker is searching the next position or the next product. The "Pick" symbolises the physical movement, when the products are moved from the position to the UL. The "Setup" is a set of preparation tasks (e.g, pallet pick-up, administration etc.), which should be done at the beginning of the order picking of a UL [68].

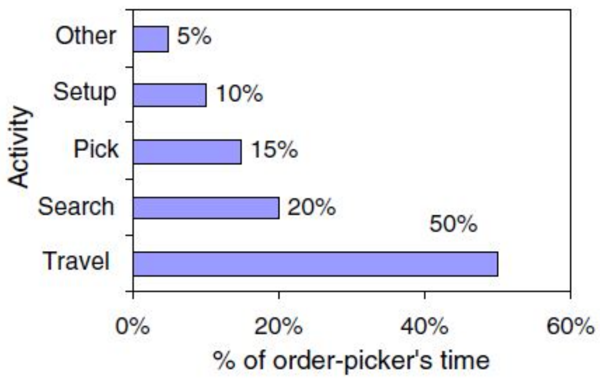


Figure 2: Time distribution of the order picking movements [68, 42]

While travelling is an emphatic element of the order picking, minimisation of its time needs is essential to improve the order picking effectiveness. The objective of the order picking routing optimisation is to sequence the records of an order picking list, where each record defines an order picking position. The departure and the arrival depots of the picker can be different. The right order picking routing minimises the travelling time to support the shortest order picking lead time. The travelling time depends mainly on the distances. However, several further aspects could influence the travelling time, for example the traffic, the different allowed speed at corridors, the stability of the moved UL. The unstable UL forces the picker to reduce the speed.

Furthermore the routing sequence has an impact on time needs of further movements. For example the picking sequence can influence the picking time, if the defined sequence requires UL reconstruction during picking.

2.1.1.1 Travelling Salesman Problem (TSP)

Based on the literature, the order picking routing problem can be defined as a special case of the Travelling Salesman Problem (TSP), which is defined by Lawler et al. in

1985. In the case of TSP, the salesman starts from the home city, a number of cities have to be visited exactly once and the salesman returns home. The distances between the cities are known. The aim of the salesman is to visit the cities and to minimise the travelled distance. In the case of the order picking routing problem, the salesman is the picker, the cities are the order picking positions and the depot is the departure and arrival position [44, 68, 42].

Koster et al. visualised the order picking routing problem by Figure 3. The left side of Fig. 3 shows an order picking task on a warehouse layout, and the right side illustrates its graph representation. While the black points are obligatory positions where the picking movements will happen, the white points are optional. The graph representation sample shows the crossroads as optional white points [42].

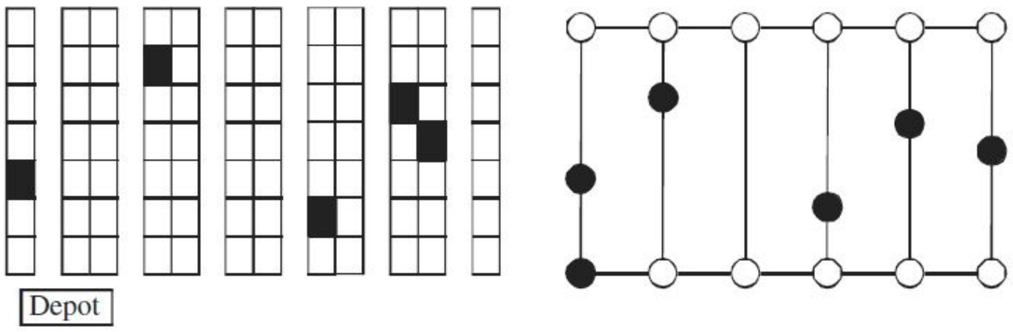


Figure 3: Order picking routing illustration [42]

Although the basis of the classical TSP and the order picking routing problem are the same, there are some differences: [42]

- The picker can visit the picking positions more than once.
- There are positions in the warehouse which can be visited, but it is not obligatory.

The visit means, the picker is working on a position or the picker just travel in front of the position. The described differences classify the order picking routing problem as a Steiner Travelling Salesman Problem. In the case of Steiner TSP, there are positions which do not have to be visited, but allowed, and the positions can be visited more than once [42].

Although the (Steiner) Travelling Salesman Problem is in general not solvable in polynomial time, Koster et al. summarised some dedicated routing heuristics, which are able to result in acceptable solutions for single-block warehouses without time-consuming optimisation algorithms. The warehouse block is a storage area of the warehouse layout, which is bounded by cross aisles or walls. The disadvantage of the optimal routing heuristics has been summarised as follows: [42]

- The optimisation algorithms might not be available for every layout. Specific infrastructures and operation control policies could exclude applying optimisation algorithms.
- The optimal routes could be illogical for the pickers
- While the optimisation algorithms can not consider the traffic jams, dedicated heuristics can reduce aisle congestions. For example, it is possible to change the traffic direction in the case of S-shape heuristics.

Koster et al. summarised the dedicated routing heuristics of one block warehouses based on Roodbergen's research. Figure 4. visualises the relevant dedicated heuristics (S-shape, Return, Mid-point, Largest gap, Combined) and compare them with the optimal solution on the same basis.

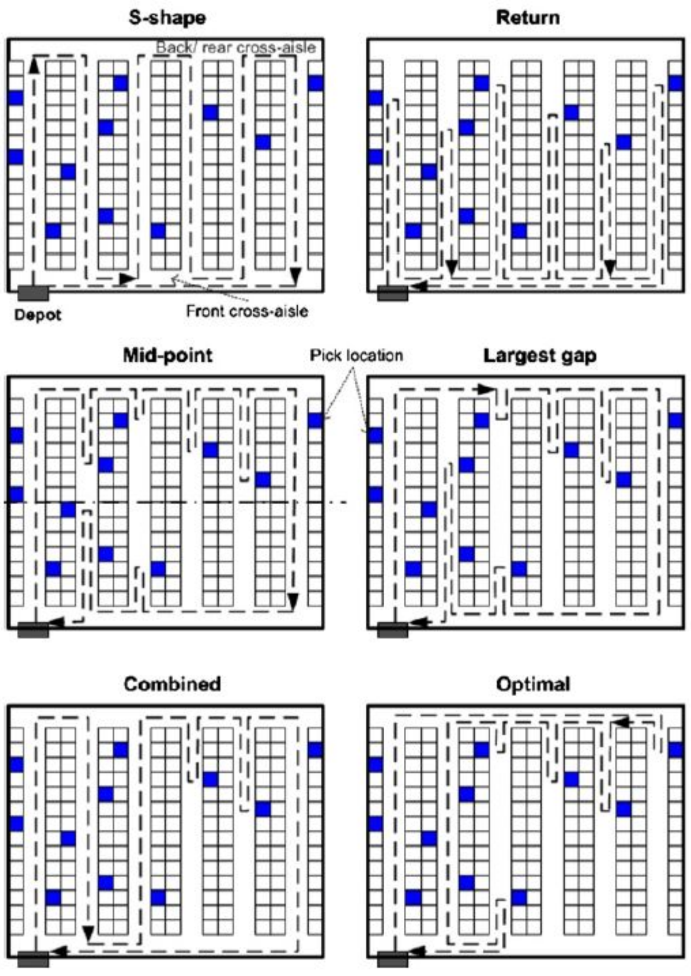


Figure 4: Dedicated routing heuristics [57, 42]

In the case of the S-shape heuristics, the picker travels through entirely each aisle, where at least one picking position is defined. The aisles, where the picker has not worked are neglected. The picker returns to the depot, when finished at the last aisle. The Return strategy works similarly like the S-shape, but the picker leaves the aisle on the same point where entered. In the case of the Mid-point heuristics, the warehouse is separated into an upper and a lower part. The picker travel through the first and the last aisle. The further aisles are visited via the upper and the lower entrance depending on where the picking positions are. This strategy is effective against the S-Shape, when the number of picking positions per aisle is low (e.g. one position per aisle on average). The Largest gap works similarly as the Mid-point heuristics, except the warehouse separation. The picker enters the aisle to reach the biggest gap between the visited and unvisited positions within the aisle. It can result in a return strategy in the case of certain aisles. The Largest gap heuristics generally performs better than the Mid-point heuristics, but its implementation is more complex. In the case of Combined strategy the picker travels through the first and the last aisle and the right routing strategy is defined aisle-by-aisle by using dynamic programming [42].

Koster et al. summarized results, which highlighted, that the combined heuristics could perform the best results or the optimisation algorithms are barely better than the dedicated heuristics. Therefore Koster et al. offer dedicated heuristics for routing optimisation.

However, if there are special conditions for picking, the effectiveness of dedicated heuristics will decrease. Special conditions can be for example the regulated picking sequence based on product stacking attributes or order characteristics effect on the picking sequence. I collected several further conditions, which will be discussed in Section 3.1.

Theys et al. compared the dedicated routing heuristics with TSP heuristics and local search algorithm in the case of multi-parallel-aisle (multi-block) warehouse. Theys et al. adapted the Lin–Kernighan–Helsgaun (LKH) TSP heuristic for order picking and applied the dedicated heuristics for initialisation as hybrid alternatives. The LKH TSP heuristics has been developed by Helsgaun. Further algorithms has been defined based on 2-opt local search method, which has also been initialised with dedicated heuristics. 2-opt is a commonly used local search operator, which removes two edges (a,b) and (c,d) from a given solution and replaces them with the two other edges (a,c) and (b,d). Exchanges are accepted when it results in shorter distances [67, 34].

Theys et al. concluded the following facts based on evaluation of the defined algorithms on the same basis [67]:

- The state-of-the-art LKH heuristics for TSP can be a well-performing solution for order picking routing. It can decrease the distance up to 47% compared to the dedicated heuristics. The longer calculation time (average = 0.25 sec, maximum = 3.56 sec) can be acceptable because of the considerable route length and operation time reduction.

- It is not necessary to combine the LKH TSP heuristics with dedicated heuristics, because these algorithms did not realised significant distance reduction.
- The combination of simple 2-opt local search operator and dedicated heuristics can results in 10–42% average distance savings besides the dedicated heuristics. It highlights, that time-consuming heuristics seems unnecessary in the defined cases.

We can realise based on Theys et al.'s conclusion, any optimisation method (heuristics, local search) can result in a significantly lower travelling time than the dedicated heuristics in the case of multi-block warehouse. However Theys et al. do not recommend time-consuming heuristics, they did not considered special sequencing conditions, which can make the heuristics solutions necessary.

Scholz et al. defined a mathematical programming formulation for Single-Picker Routing Problem, which is applicable for multi-block warehouses. The research considers the cross-aisles and the movements within the aisles. It highlights the necessity of considering the changing positioning of departure and arrival position and the importance of the simple and transparent routing from the picker point of view. The model assumed, that the setup, the searching, and the picking times are constant [62].

However, if the UL reconstruction is allowed during order picking, the picking time will depend from the sequence of the previously picked products and can not be constant.

Földesi et al. highlighted, that more realistic TSP solutions would be necessary, because the travelling costs or times are rarely constant and predictable. The research offers an eugenic bacterial memetic algorithm for the TSP, which applies fuzzy coefficient for the cost definition [32].

It highlights, that applying evolutionary algorithms for TSP could be necessary, depending on the complexity of the examined problem.

However, most of the investigations in the field of the order picking routing problem ignore the product stacking and the pallet loading aspects, Molnár and Lipovszki highlighted the importance of a well sequenced order picking list to support well structured and stable ULs to avoid product damages. They realised, that inaccurate and simplified order picking lists may require UL reconstruction during order picking, which reduces the picking efficiency and increases order picking lead time. The research highlighted, that the handled products must be categorised based on weight, size, shape, packaging and loading attributes. The developed routing optimisation algorithm minimises the routing distance and the difference from the defined sequence of product classes [49].

Although the proposed routing algorithm already considers product stacking aspect and gives a reliable solution for effective picking of stable ULs, further aspects should be considered. Besides the product parameters, the order picking list itself (e.g. low and high ordered quantity) can also influence the picking classes, which requires a more flexible and more complex sequencing rule definition.

Veenstra et al. considered the handling costs and the truck load reconstruction during transport in the case of pickup and delivery TSP. The pickup position is, where the products are loaded into the vehicle. The delivery position is, where the products are unloaded from the vehicle for the client. The handling cost during delivery is increased, when the necessary product is not the next on the vehicle. The results highlighted, that applying reconstruction handling cost into pickup and delivery TSP is necessary to minimise the operation time [69].

However, the research does not concentrate on order picking routing problem, applying reconstruction into TSP highlights the necessity of my research.

I realised based on my order picking routing state of the art research, that loading and product stacking aspects are not applied with the right weight and comprehensively in order picking routing algorithms. However, it would be necessary to minimise the order picking operational time.

2.1.1.2 Routing and Loading

Arranging the products on the pallet relates to the Container Loading Problem algorithms (CLP). The Container Loading Problem algorithms are responsible for assigning three-dimensional small products to three-dimensional rectangular large objects (i.e., truck, containers, pallet). Its aim is to hold the basic geometric feasibility conditions and reach the defined problem specific objective function [13].

The basic geometric feasibility conditions are [13]:

- the small items are positioned within the container
- the small products do not overlap

Bortfeldt and Wäscher collected and structured the main objective functions and problem types of the CLP, where Pallet Loading Problem (PLP) is a maximisation problem. It is responsible for packing the maximum number of identical rectangular boxes onto a rectangular pallet [2].

Shiau and Lee (2010) solved the multi-container loading problem and defined the order picking sequence based on loading aspects. The proposed solution has 3 steps. The container selection, which defines the proper container(s) for the order based on geometric dimensions to reduce packing costs. The loading configuration step is responsible to avoid the product overlapping within the container, to orient the products within the container, and to ensure, that the products fit within the container. The loading sequence is defined based on the arrangement of the products within the container. It takes into consideration the geometric parameters of the products and the container. This loading sequence defines the picking sequence [63].

Shiau and Ma (2014) proposed an order picking heuristic algorithm for economical packing in the case of on-line shopping companies. This model applied similar loading aspects for defining the container contents and inner structure. It applied the nearest neighbour TSP heuristics for sequencing each layer as a container. It defines the nearest position from the actual position as next step [64].

Although the research defined the picking sequence and the routing besides the geometric loading parameters, the proposed models avoided the possible picking sequencing limitation (e.g. fragility, stacking possibilities etc.).

Based on my state of the art research I can highlight, that many researchers have attained valuable results in the fields of Pallet Loading Problem (e.g., [46, 43, 5]), but there is a lack of harmonising the order picking routing algorithms with the PLP solutions.

Although the solutions for vehicle routing problem (VRP) consider loading aspects, there is a lack of order picking routing problem specified VRP solutions with loading constraints. In the case of VRP a given vehicle fleet should perform the given transports. The main objective is the routing costs minimisation. Several VRP sub-problems and the main loading constraints have been summarised by Pollaris et al.. The defined loading constraints (geometric dimensions, fragility, orientation, stacking, and priority) are mainly product parameters, but in the case of order picking routing problem further aspects would be necessary. These aspects will be discussed in Section 3.1. The order picking routing problem would be a special case of the VRP with loading constraints when one picker has one set of positions. The positions have to be visited and the products should be arranged on a pallet to build a stable UL [56, 18].

2.1.2 Storage Location Assignment (SLA)

The aim of this section is to highlight the usually applied SLA methods and the considered assigning parameters.

The SLA methods are responsible to define the storage position of a UL during storing in process and to allocate products to picking positions to support the effective order picking. It has a considerable impact on the order picking travel distance and time, the effectiveness of the picking sequence, and the necessity of the UL reconstruction [42].

Koster et al. summarised the main SLA methods, which are usually applied in research and in practice. The *random* storage method selects randomly a position from the possible free locations with equal probability. It results in high storage capacity utilisation and increased travel distance. The *closest open location* storage is applicable, when the warehouse workers can choose a free position during storing the UL into the storage zone or during order picking replenishment by themselves. Another solution is to store the products on fixed positions by the *dedicated* storage strategy. Its disadvantage is the possibly lower space utilisation, because the storage position might be reserved when the dedicated product is out of stock. However, in the case of dedicated order picking positions it can support the effective order picking. The order picking sequence could be transparent and logical for the pickers. The storage positions can be handled by another strategy. Koster et al. highlighted, that dedicated storage can support the order picking sequence, if the products have different weight. Dedicating the products to positions by increasing sequence of weight can help pick first the heavy and finally the light and sensitive products. This highlights the necessity of my research, but the example applies only product parameters. The *full turnover* storage allocates products to positions based on their turnover or ordering intensity. The fast moving or frequently picked products should be allocated near the departure and arrival position and the rarely handled products should be stored farther to minimise the order picking distance [42].

The *class based* storage combines some previously mentioned methods. The products are classified and dedicated based on any aspects (e.g. turnover and product parameters). The same classes are grouped and allocated in a coherent area. One of the usually applied class based allocation policies is the combination of the full turnover and the dedicated storage. The products are classified based on Pareto's method into A, B and C classes and dedicated into class based areas. The A products are generally 20% of the sum number of products and give 80% of the demands. The B and C products are less and less frequent. The usually applied Pareto or ABC analysis can be defined based on several aspects (e.g, ordered quantity, ordering frequency, volume, etc.) depending on the aim of classification. Figure 5 and Figure 6 illustrate some class based storage implementations [42, 55].

Theys et al. examined the previously described (Section 2.1.1) routing alternatives in the case of random and volume-based centralised SLA. The volume-based SLA dedicated the most demanded products in a centralised area near the departure and arrival position. The results clarified, that the volume-bases SLA realised lower distances in the cases of each routing alternative [67].

Dijkstra and Roodbergen proposed a dynamic programming solution for class-based SLA. It considers the applied routing method (S-shape, Return, Largest-gap, Midpoint), the length of picking lists, and the demand distribution of classes within the picking list. The proposed solution offers optimal allocation for S-shape and Return routing, but the route length estimation and the SLA methods do not consider the product stacking aspects and necessity of UL reconstruction [28].

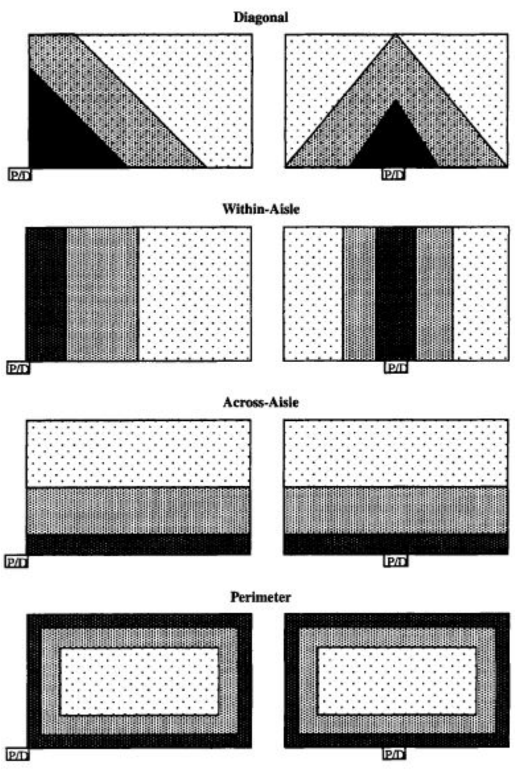


Figure 5: The possible class based storage implementations [55]

The *family grouping* SLA strategy considers the relations of products based on ordered quantity, ordering frequency or product association. For example the products which are usually ordered together should be stored near to each other. In the case of family grouping strategy the previously described solutions can be combined with considering for example the correlations of classes [42].

Instead of the usually applied quantity- and frequency-based family grouping SLA, Chuang et al. defined a family grouping SLA model based on item association. The item association is defined based on the chance that the products occur together in the same picking list. The research concentrates on a picker-to-parts, pick-by-order warehouse, which is working with small lot sizes, diversified contents, and short response times. First, the model is clustering products into groups based on calculation of the correlation between products. Then storage sequence of groups and products are defined based on average order frequencies. Chuang et al. proposed a robust heuristic method for the problem in 2016, which provides results with maximum 6,1% difference from the previous version, but in a far more effective way [23, 22].

Pan et al. highlighted, that traffic jams in the picking aisles during picking should be considered during SLA. The research defined a heuristic approach, which considers the travel distance and the picker blocking time during SLA to minimise the order fulfilment time. It can determine the number of aisles needs to be balanced to minimise the blocking and travel time [53].

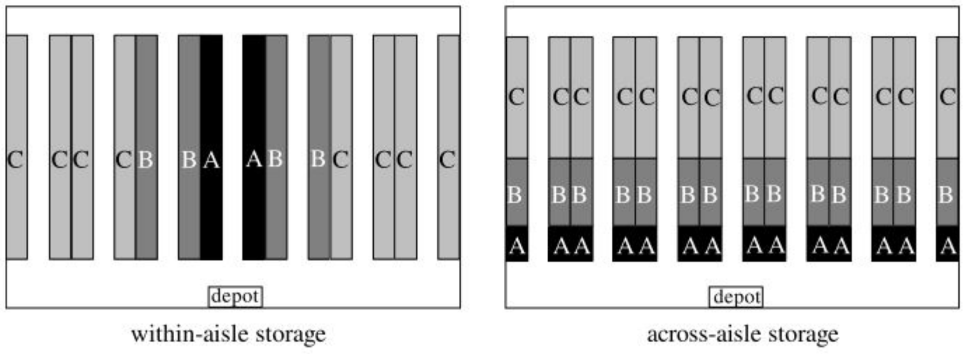


Figure 6: The most common class based storage implementations [42]

Many further solutions have been defined for harmonising SLA and routing to decrease the routing distances and times (e.g., [48, 19]). Therefore, most of the research works considered product parameters (turnover, ordered quantity, ordering frequency, storage space requirement), item association or picker blocking. However, while the physical product parameters (dimensions, weight, packaging) and product stacking attributes influence the physically possible picking sequence in order to build stable ULs, researchers rarely take into account these aspects during SLA and routing optimisation.

2.1.3 Layout

Koster et al. highlighted the main decisions in the field of warehouse layout. Its most important aim is the routing and material handling cost minimisation. Figure 7 visualises the main internal warehouse layout related questions well. The length, width, and number of aisles depend on the storage capacity requirements and the applied material handling technology. The cross aisles define storage blocks and let the pickers to change aisle with shorter travel to minimise distances. However, the frequent cross aisles decrease the storage capacity. The number of storage blocks is defined based on for example the needed separated functions (e.g, different partners, sensitive products), routing optimisation, order picking strategy (e.g. zoning) requirements, and specificity of the building [42].

The most important question for my research is the number and position of the departure and arrival (DA) position (depot). It has a considerable effect on the routing distances and times, because the picking of a UL usually starts and finishes here. The routing algorithms should count the picking position sequence from this point. The SLA should also consider the DA position. For example when the products are allocated based on ordering frequency, the usually ordered products should be stored near the DA position.

Theys et al. examined centralised and decentralised DA position positioning based on the previously described (Section 2.1.1) routing alternatives. The results highlighted, that

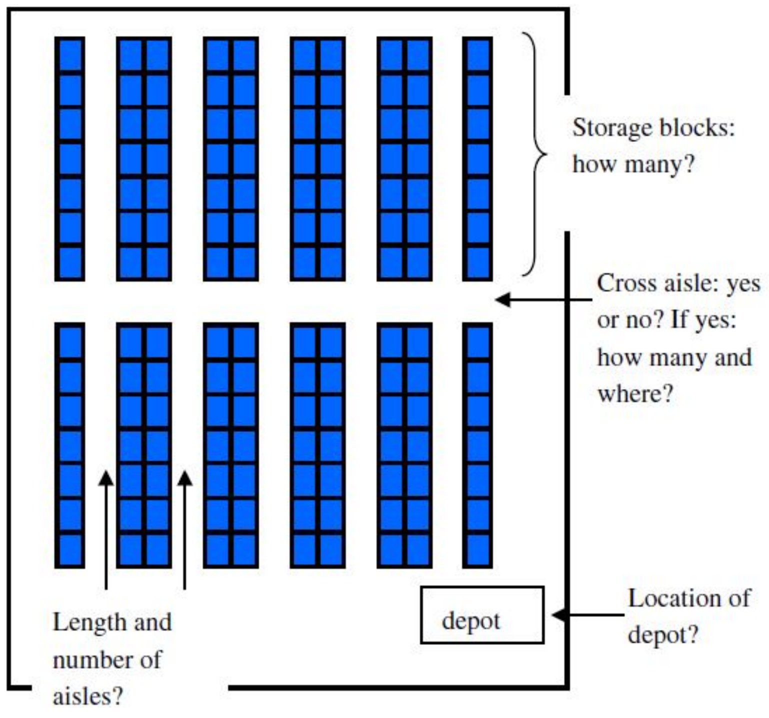

Storage blocks: how many?
Cross aisle: yes or no? If yes: how many and where?
Location of depot?
depot
Length and number of aisles?

Figure 7: Decisions of layout design [42]

the dedicated routing heuristics resulted in lower distances in the case of decentralised DA positions, because the routes usually start in the first aisle. The defined routing heuristic has similar results in the case of centralised and decentralised DA position [67].

The results highlight, when the picking sequence is determined by any factor (e.g. dedicated heuristics or product stacking parameter) the positioning of the DA position has a significant impact on the routing distance.

2.1.4 Zoning

The order picking area can be divided into zones based on product parameters or operation controlling aspects. For example products, which require different storage temperatures should be stored and picked in separated, and temperature controlled zones. Huge picking areas should be separated for zones, when the pickers are assigned to a zone for picking the separated part of the order. The advantages of zoning are transparent smaller area for the picker and reduced traffic jams. The main disadvantage of zoning is that picked ULs must be consolidated and the transport units should be defined after picking. The zone picking can be controlled sequentially. The picker forwards the picking list and the picked UL to the next zone, where the next picker will continue. In the case of parallel picking the UL consolidation is more important, because the pickers perform the separated picking list in the assigned zones independently [42].

While my research is focusing on order picking within one zone when the order picking list is given for the zone, zoning decisions and optimisation are not connecting directly to this research. However, the complexity of routing optimisation has been examined (Section 3.2.1) in the case of separated pallet loading factor based zones as a preliminary research.

2.1.5 Defining the order picking lists

The order picking lists are defined based on the received orders. However, the orders usually should be batched or separated based on the different ordered volumes to increase the efficiency of order fulfilment. The order picking list definition is influenced by the ordered volume as follows:

- If the ordered volume of the received order **fits** for one UL, then the order picking list is simply going to contain the records of the order.
- If the ordered volume is significantly **less** than one UL and the UL utilisation would be low, then a couple of orders should be **batched** to define a picking list.
- If the ordered volume is **more** than one UL, then the order should be separated for UL based picking lists to support the pickers. It can be defined as a **bin packing** problem.

Furthermore, if the warehouse operates separated order picking zones, then besides the volume based decisions the lists should be separated for zones, too.

The order characteristics (e.g, product types, quantity, volume), the zone specifications, the SLA, and the picking rules strongly influence the order picking list definition in both batching and bin packing cases.

Although the loading possibilities have also an impact on the batching and bin packing decisions, these are actually out of the scope of my research. While my research assumes previously defined order picking lists, the batching and the bin packing are mentioned marginally.

2.1.5.1 Batching

Koster et al. highlighted, that batching small orders with defining picking list as a set of orders has a potential for travel time reduction. Orders are allocated to a picking list while those are fit to a single picking tour of a UL. The most common order batching solutions are the proximity batching and the time window batching. The proximity batching assigns orders to a batch based on proximity of the picking positions. Its aim is to minimise the picking distance of a batch. In the case of the time window batching the orders received in the same time window are batched [42].

Since the general aim of the batching is the travelling time minimisation, several research works have been made in the field of batching and routing synchronisation (e.g. [54, 52, 47]).

When the product stacking and the possible picking sequence have an impact on the routing, these factors should be considered during batching also to define an effective order picking on the pallet loading point of view.

2.1.5.2 Bin packing

Separating the big orders for ideal picking lists to minimise the order fulfilment time is generally a bin packing problem. Separating the order is necessary to support the picker during picking. If the whole order is assigned for the pickers, then they have to use their brain to separate the long picking list for ULs with considering several factors (e.g, number of ULs, travelling distance, UL stability).

According to Bortfeldt and Wäscher's state of the art review in the field of Container Loading Problem (CLP), bin packing is a minimisation problem. It is responsible for assigning strongly heterogeneous products into a minimum number of containers [13]. Several research works have been done in the field of Bin Packing problem (e.g. [60, 26]). The research works consider 2D, 3D, homogeneous, heterogeneous products and containers. The stacking aspects have been considered for example by Saraiva based on the weights of the products. The heavy products are allocated on the bottom of the stack.

While the sequence of the products is assigned within a container, the bin packing algorithms should consider the possible UL building rules, not just the weights.

2.2 Optimisation

Several tasks and problems are known and managed in practice which have a huge amount of possible solutions or synchronisation of several related aspects are necessary. Defining the right solution for these tasks would be hard for the pickers without any optimisation support.

To apply a reliable solution for an operative problem, it is necessary to find the appropriate optimisation methodology for the given problem depending on its complexity. The problem complexity mainly depends on the search space and the available time for the optimisation. The search space defines the number of possible, valid solutions for the problem. A huge dimensional search space causes an increased complexity of the problem, because of the evaluation of each possible solution is impossible within the available time window. The time window for the optimisation is usually defined based on the possible application. How much time the algorithm has to provide a solution for the problem.

The optimisation problems can be divided into two classes: continuous and discrete optimisation problems. The continuous optimisation problems have uncountable (uncountably infinite) possible solutions, uncountable search spaces. When the search space is uncountably infinite, finding exact solution in general cases is nearly impossible. The continuous optimisation algorithms generally concentrate on finding an acceptable solution for the given problem. The discrete optimisation problems have countable (finite or countably infinite) possible solutions, countable search spaces. It is often referred as combinatorial optimisation, because the search space is usually defined based on a combinatorial mathematical method [3].

Even in the case of the discrete problems there are computationally complex problems (so-called NP-hard problems [33]). In this case the search space is extremely growing, a "combinatorial explosion" is happening, when any parameter of the problem is growing. In the case of a huge dimensional search space or computationally complex NP-hard problems it is impossible or it would take too long time to find an exact solution. This is the reason, why the general objective of the optimisation algorithms is to define quasi-optimal solutions for the given problem. The quasi-optimum is an acceptable solution, which generally requires less computational time, than finding the exact optimum. Finding the best solution is rarely necessary or sometimes impossible in realistic time, quasi-optimum is usually enough for the effective operations [3].

Based on the nature of the logistics optimisation problems, most of them are defined as a discrete optimisation problem. However, the dimensions of the search space can be different, which may require a different detailed optimisation methodology. The literature applies three main optimisation methodologies for the different search spaces:

- Exact (brute force) algorithms
- Dedicated heuristics
- Meta-heuristics - Evolutionary algorithms

The exact (brute force) algorithms are applicable, when the search space is small. The algorithm has time to calculate the optimal solution or evaluate every possible solution to find the optimum. The aim of these algorithms is to find the acceptable solution from the search space [31].

The dedicated heuristics are usually applied, when it is impossible to evaluate each possible solution and it is possible to decrease the search space based on some rules. This method might won't find the optimum, but it is able to provide a reliable solution without any complex optimisation methodology [31].

The nature inspired evolutionary algorithms as one branch of soft computing, meta-heuristics techniques are able to reach optimal or acceptable solutions for complex problems within short available time window. Evolutionary algorithms are also applicable, when the analytic description of the problem is only partial or the nature of the system is partially known. These are suitable for global optimisation in the case of non-linear, high-dimensional, multi-modal, and non-continuous problems [14].

As part of my state of the art research I summarise the usually applied related algorithms and the relevant evolutionary algorithms:

- Simulated Annealing
- Genetic Algorithm
- Bacterial Evolutionary Algorithm
- Memetic Algorithm
- Swarm intelligence techniques

The basis of Simulated Annealing (SA) method is defined by Kirkpatrick et al. for TSP [40]. Its main feature is the allowable choice of actually weaker solutions with a decreasing probability in order to move the algorithm out from the local optimum [40]. This well-known optimisation algorithm has often been used to evaluate developed algorithm results on a standard basis [24]. Besides the classical application, it can also be used in a population-based manner, when the defined weak individuals are accepted with a decreasing probability.

The Genetic Algorithm (GA) is one of the most widely used evolutionary methods. In this algorithm, similarly as in all evolutionary and swarm based optimisation algorithms, one individual represents one possible candidate solution to the given optimisation problem. The crossover between chromosomes, the random mutation of individuals, and the ensured higher chance for the valuable individuals to survive are the key points of GA. The initialisation creates the given number of individuals in the search space randomly. After the evaluation of the fitness value of each individual, a given number of individuals is selected, where the fitter individuals have higher probability for selection. The selected individuals are paired and during the crossover randomly selected gene(s) are selected and changed between the previously paired individuals. The genes of the resulted new individuals are mutated with certain probability. The defined new individuals overwrite some selected individuals during the substitution step. The individuals to be overwritten are usually defined based on the fitness values. Usually the weaker individuals will be overwritten by a new individual. The selection, crossover, mutation, and substitution are repeated in the iteration loop [35].

The Bacterial Evolutionary Algorithms apply bacterial mutation and gene transfer operators inspired by the bacterial evolutions. During the bacterial mutation each bacterium is randomly mutated using several clones (copies) of the bacterium. If a mutated clone achieved a better fitness value after the mutation, then it overwrites the original one. The gene transfer operator divides the population into two groups based on the fitness values. The individuals with higher fitness values are dedicated to the superior group and the weaker individuals to the inferior group. Then superior and inferior bacterium are paired and randomly defined genes are moved from the superior bacterium to the inferior one. The bacterial mutation and the gene transfer are repeated in the iteration loop [51].

The evolutionary algorithms are improvable by local search approaches, which might improve the performance of the evolutionary algorithm and help them find the global optimum faster. The combination of evolutionary and local search methods are referred to as memetic algorithms, which are often applied to provide an approximate solution for NP-hard problems (e.g., [36, 66]) [50]. Botzheim et al. proposed a new kind of memetic algorithm based on the bacterial approach, the Bacterial Memetic Algorithm (BMA) [16, 32].

BMA has already been utilised in several combinatorial optimisation problems such as TSP [32], robot path planning [17], feature selection problem [15, 74]; and in continuous optimisation tasks such as fuzzy rule base extraction [16], building energetics [25], robot locomotion [59].

Swarm intelligence techniques are inspired by the observed behaviour of the crowd in the nature (e.g. ant, bee colony). Each individual (member of the crowd) tries to find better and better solutions for the problem by evaluating the environment and applying the own and the shared experiences. As an optimisation method, the individuals can represent a possible solution for the problem [30].

One of the famous methods is the Particle Swarm optimisation (PSO), which models the dynamics of a multi-particle physical system. The particles are continuously moving within the search space based on their local and the global best places. If the individual finds a better solution than its local best, than it will be its starting position in the next iteration. If the individual finds a solution, whose fitness value is higher than the global best, than this solution will be the global best solutions. At the end of the iterations the global best place (solution) will be the quasi-optimum [39].

Artificial Bee Colony algorithm (ABC) defines three groups of bees: employed bees, onlookers, and scouts. It is assumed that there is only one employed bee for each food source. It means, that the number of employed bees in the colony is equal to the number of possible food sources. Employed bees go to their food source and come back to the hive and dance on this area. The employed bee whose food source has been exhausted becomes a scout and starts to search for finding a new food source. Onlookers watch the dances of employed bees and choose food sources depending on the dances [38].

These swarm optimisation methods are working well on continuous optimisation, however, they are not prepared for solving combinatorial optimisation problems [30].

Ant Colony optimisation (ACO) models the ant behaviour during food searching within a colony. The ants define the paths from their nest to the food sources partly randomly, partly based on their intuitions and partly based on the pheromone trails of other ants. The ants leave pheromone trails where they go. The pheromone uniformly evaporates in time, however the shortest path will be visited the most frequently and it has the strongest pheromone trail. The ants visits the route having the strongest pheromone trails with the highest probability. In the case of long terms the whole colony will be transporting food on the shortest paths. The ACO can be applied well when the problem is formulated properly [29].

There are some other well-known nature inspired optimisation algorithms. Differential evolution (DE) can be considered as an evolutionary algorithm and also as a swarm intelligence algorithm. In differential evolution the mutation of an individual happens based on the difference of the locations of two or more other individuals [65]. Artificial Immune Systems and related algorithms such as Clonal Selection algorithm are inspired by the principles and processes of the immune system [27, 37]. Many other evolutionary and swarm optimisation algorithms can be found in the literature. Interested readers are referred to [30, 45, 71, 72].

Chapter 3

Own results

3.1 Order Picking Routing Problem based on Pallet Loading Feature - OPRP-PLF

My state of the art research highlighted the typical fields and schemas of the order picking routing and SLA research.

Most of the research apply dedicated routing heuristics, however, the effectiveness of optimisation has been proved in the case of a bit more complex systems (e.g, multi-block layout, loading aspect). Furthermore, meta-heuristics algorithms have also been applied for TSP because of the more flexible algorithm definition and the possibly lower computational time.

Although VRP research has considered loading aspects, there is a lack of order picking routing problem specified VRP solutions with loading constraints. Furthermore, despite the fact, that the VRP and some order picking routing research works have considered some product parameters (e.g weight, dimensions, fragility, orientation, stacking, and priority), more aspects (e.g, order characteristics, layout, SLA) should be synchronised in the case of order picking routing problem. However, the necessity of these aspects should be examined before algorithm implementation.

Furthermore, many solutions have been made in the field of pallet loading problem and bin packing, but there is a lack of harmonising the order picking routing algorithms with these solutions. I realised based on my industrial experiences, when these aspects are relevant then the routing algorithms should consider the possible pallet building rules. For example high quantity of the product A is not allowed to pick on top of low quantity of product B.

Besides keeping the allowed UL building rules, the order picking system should keep its flexibility with allowing the UL reconstruction during picking. However, applying reconstruction into order picking routing algorithms might increase the picking efficiency, nevertheless there is a lack of its implementation. Naturally this movement takes time, but it might result in a lower distance and a lower order picking lead time. Despite most of the research works defined the picking time as constraint, the time calculation should be inspired more by industrial experiences when it is depending on for example the picking sequence.

Many research works harmonised SLA and routing to minimise order picking lead time, because SLA has a huge impact on the sequence of the picking positions and the route distance. Most of the research considered product parameters (turnover, ordered quantity, ordering frequency, storage space requirement), item association or picker blocking. However, while the physical product parameters (dimensions, weight, packaging) and product stacking attributes influence the physically possible picking sequence in order to build stable ULs, researchers rarely take into account these aspects during SLA optimisation.

I realised based on my state of the art research and industrial experiences, that loading and product stacking aspects are not applied with the right weight and comprehensively in order picking routing and SLA algorithms. However, it could be necessary to minimise the order picking operational time. The aim of my research is to examine the necessity of the discovered contexts. The novelty of my research is to consider each influencing factor as summarised in Table 1 with examples. (The "X" shows the applied factor.)

Table 1: Novelty of OPRP-PLF [9]

	Routing	Loading	SLA	Reconstruction
Theys et al., Scholz et al.	X			
Moeller, Chan and Chan	X		X	
Molnár and Lipovszki	X	X		
Proposed research (OPRP-PLF)	X	X	X	X

3.1.1 Problem description

Inspired by my state of the art research and my industrial experiences, I defined the Pallet Loading Feature (PLF) and the Order Picking Routing Problem based on Pallet Loading Feature (OPRP-PLF). PLF is defined as logistics system attribute, which requires the right picking sequence and pallet loading method to build stable ULs and to avoid product damages. The challenges of OPRP-PLF are to minimise the order picking lead time, build stable transport units and avoid product damages, when industrially relevant but rarely discussed PLF based order picking sequencing is necessary.

As highlighted before, the PLF and the OPRP-PLF depends not only on the product attributes. The order picking list characteristics and the order picking system itself could have a huge impact on the described problem.

I collected several factors of these aspects, which should be examined and considered:

- Product attributes
  - Weight
  - Shape
  - Size
  - Packaging
  - Package fullness
  - Stacking attributes
- Order picking list characteristics
  - Ordered items
  - Ordered quantity
  - Length of picking list
  - Number of product types on the list, with different stacking attributes
  - Special customer rules for pallet loading
- Order picking system
  - Previously picked units and those sequence
  - Product assignment in the warehouse
  - Zones
  - UL making tools

Each product has several parameters, which define their physical stacking attribute and the required picking sequence. The shape and the size of the products has an impact mainly on the orientation within the UL and the volume filling of the UL. The product shape can influence the picking sequence, for example a rounded product can decrease the possibilities of the following products. The product weight has a strong connection with the fragility and the UL stability. While the heavy products usually should be positioned on the bottom of the UL, the fragile and light product usually should be positioned on the top. Packages are not only boxes but bags, cans, trays, bins or any amorphous units, whose specifications also have a huge impact on stability and stacking attribute. The stacking attribute is usually defined based on objective (measurements) or subjective (best practices) ways. It highlights the physical bearing capacity of the package, which ensure the stable UL. The package fullness could be important, when the package is weaker than the product within the package. When the package is full, then the products keep the weight, otherwise the package needs to ensure the stable UL. It has an impact on the stacking attribute of the product and influences the position of this product within the UL and the picking sequence of the ordered products.

The characteristics of orders also influence the OPRP-PLF and the right picking sequence. Its evaluation is essential during warehouse development. The ordered products' attribute and the ordered quantity influence the stacking attributes. For example a high quantity from a small and weakly packaged product can behave together like one simple box and after its picking, the picker might be able to pick further boxes. Some different products with different types of packaging can also behave stronger together, rather than separately. The length of picking list highlights the necessity of the routing optimisation. A short list usually does not require a complex and maybe time consuming optimisation. However, it is necessary to optimise the longer and more complex picking lists, which will save time for the warehouse operation. The number of different product stacking types also influences the requirements of PLF based optimisation. More and more different product types on the same picking list increase the complexity of the order picking sequence, which requires applying optimisation algorithms for OPRP-PLF. The customers usually define the expected pallet loading rules, which usually limit the possible picking sequence. For example the customer sometimes expects "sandwich ULs", which needs to pick a pallet after every picked record to separate products in the same UL. This UL type has an impact on the OPRP-PLF and sometimes changes the stacking attributes of products.

The order picking system itself has an impact on the OPRP-PLF. During order picking, the previously picked products and its quantity usually influence the possible further picked products and sequence. The dimensions, the positioning on the layout, the product allocation, and the processes of the defined order picking zones have an impact on the necessity of the OPRP-PLF. When the stacking sensitive product can be separated for zones, then complex algorithms for the OPRP-PLF are not necessary. Otherwise, if a picking zone handles several product with several stacking attribute and rules, then considering PLF is important. The picking positions of the products (SLA) have a high impact on picking distances, which influences the necessity of the routing optimisation and the UL reconstruction to reach the shortest lead time and follow the stacking rules. Sometimes a well defined SLA and a simple routing method could ensure the right UL building sequence. For example allocating the products to positions within the zone based on product weight sequence and applying S-shape routing could ensure stable ULs. When the PLF is more complex, then a more sophisticated SLA and routing will be necessary. The UL making tool (e.g., pallet, trolley) also influences the OPRP-PLF. When it is possible and effective to use trolleys with separated shelves as UL making tool, then stacking sequence is negligible. In this case, the product has less impact on each other and the picking sequence is less limited by the PLF. Otherwise, when pallets are applied, effect of the PLF on the SLA and the routing could be high.

The OPRP-PLF based development is connected to strategical, tactical, and operative decisions. Related to this research the following main challenges and questions arise regarding the different decision levels.

On a strategic level the warehouse management has to determine the long term business strategy, the main services offered, the main industry (Fast-Moving Consumer Goods (FMCG), automotive, pharmaceutical . . .), the infrastructure requirements, and development goals.

On a tactical level several decisions should be made regarding policies and algorithms. It is important to define an ordering policy, as it is allowed for the customers to purchase order (minimum quantity, SKU, package, ordering time window, etc.). The handled product types and those possible packaging solutions should also be determined. The product attributes will be one input to define the relevance of OPRP-PLF, which must be examined on tactical level as an initial step of algorithm development. Warehousing algorithms (storing in, storing out, replenishment, routing) should be developed, which hopefully will support the operational decisions. If the OPRP-PLF is relevant, the warehousing algorithms should take into consideration the OPRP-PLF with the right weighting. The SLA should also be determined on a tactical level and continuous re-engineering is necessary based on seasonal or periods of changing demand.

On the operational level the warehouse management has to make several decisions hopefully supported by algorithms. When the orders arrive at the warehouse and the order picking tasks are defined, it is necessary to determine: does the ordered quantity fit into one UL or how many UL will be necessary? It is a complex and important question of how the ordered products will be separated to ULs. The optimised order picking routing of the rightly defined UL picking lists should result in the shortest order satisfaction lead time and result in stable ULs. The routing optimisation is strongly connected with the SLA and with the PLF. Due to complex OPRP-PLF, in the case of a well designed SLA and routing algorithm, the shortest picking distance might not result in the shortest lead time, because the picker might have to spend time on UL reconstruction during order picking. The necessity of reconstruction can be caused by higher or lower ordered quantity, as it is assumed during SLA, because different amounts of product can behave differently on the UL. In this case a longer distance might result in shorter lead time because of less pallet loading time. On an operational level the routing algorithm should decide, how to reach the shortest lead time. The possible solutions are to collect products in the right PLF based sequence and walk more or pick with shorter routing and spend time on redesigning the contents of the UL when it is necessary. The best choice depends on the SLA, the time requirement of movements, and the length and contents of the picking list. The increasing frequency and time requirement of pallet loading and reconstruction can highlight the decreasing efficiency of SLA and the necessity of its re-engineering. The well-defined Key Performance Indicators (KPIs) can highlight the necessity of tactical decisions or re-engineering.

3.1.2 Modelling the pallet loading possibilities

Warehouses are faced with the complex position sequencing problem of order picking lists when PLF is relevant during the order picking of customer orders. Several VRP, bin packing and PLP algorithms have been developed, which usually require appropriate data regarding every product (dimensions, weight, stacking attribute, etc.). However, the warehouses usually lack the exact product parameters, well-defined stacking constraints, and algorithmically handled customer requirements. When the exact data is available, but rules are changing continuously, the warehouses require frequent measurement and algorithm upgrades. Without updates the problem specific algorithms won't be able to give constant performance and the pickers should spend more and more time to consolidate the picking sequence based on best practices. In this case the algorithms and their input should be approximated to the real nature of the system in order to support the operational decisions.

My algorithms for OPRP-PLF require simplified input, which is definable for every warehouse without a measurement of each product. Based on known, easily measurable, and rarely changing information it is possible to classify the products and every order line. My model can apply the warehouse workers' best practices and experiences, which are essential and valuable inputs to the classification. The defined Pallet Loading Classes (PLC) have PLF based logical connections (i.e., we should not put heavy goods on fragile products), which will be the basis of the OPRP-PLF algorithms. The mentioned classification process will be described in the following sections step by step.

3.1.2.1 Classification of the product parameter based classes

The products are grouped based on physical product parameters, which has different stacking attributes (Eq. (1)). The usually considered factors of Product Classes (PC) are the SKUs, the packaging solution, and the product attribute. Equation (2) shows a possible industrial example [6].

$$PC = \{A, B, C, D\} \tag{1}$$

$$PC = \{PlasticBin, CartonBox, SmallBox, Fragile\} \tag{2}$$

3.1.2.2 Classification of the product and order parameter based classes

The defined PCs are specified based on order parameters to define the Product and Order Parameter based Classes (POPC). If it is necessary I separate PCs into further classes. The high quantity of the same product usually has different stacking parameters. For example element B (CartonBox) of the PC set is separated into High Quantity (HQ) and Low Quantity (LQ) elements (Eqs. (3) and (4)). In this case, if the ordered Quantity (Q) is equal to or higher than 4 then the order line is classified as B_{HQ}. Four carton boxes – which are stored in a full layer on the pallet – can be more stable on a UL than only one carton box. If the ordered Quantity (Q) is smaller than 4, then the order line (r) is classified as B_{LQ}. Equations (5) and (6) show an example of the CartonBox class [6].

$$B_{HQ} \in POPC \mid (r_{PC} = B) \wedge (Q \geqslant 4) \tag{3}$$

$$B_{LQ} \in POPC \mid (r_{PC} = B) \wedge (Q < 4) \tag{4}$$

$$CartonBox_{HQ} \in POPC \mid (r_{PC} = CartonBox) \wedge (Q \geqslant 4) \tag{5}$$

$$CartonBox_{LQ} \in POPC \mid (r_{PC} = CartonBox) \wedge (Q < 4) \tag{6}$$

3.1.2.3 Classification of the special product and order parameter based classes

The Special POPCs (SPOPC) are defined with consideration of previously picked units and their sequence (Eq. (7)). The picked products on the pallet form a physical structure, which influences the choosing of subsequent products. For example, it is possible to pick one layer of small boxes, after one layer of small boxes but the third layer of small boxes would destabilize the UL, so in this case it is forbidden to pick one layer of small boxes after two layers of small boxes (Eq. (8)) [6].

$$S \in SPOPC \mid (POPC_{t-1} = Y) \wedge (POPC_{t-2} = Y) \tag{7}$$

$$\begin{aligned} &SmallBoxSmallBox \in SPOPC \mid \\ &\quad (POPC_{t-1} = SmallBox) \wedge \\ &\qquad (POPC_{t-2} = SmallBox) \end{aligned} \tag{8}$$

Where t is the actual picking step, $t-1$ is the previously picked POPC, and $t-2$ is the last but two picked POPC.

3.1.2.4 Defining the PLFDM

PLF based Decision Matrix (PLFDM) models the PLF based sequencing logic. The predecessors (rows) are the elements of the Pallet Loading Class (PLC) set, which are the union of POPC and SPOPC sets. The successors (columns) are the elements of the POPC set (Eqs. (9) and (10)) [6].

$$PLC = POPC \cup SPOPC \tag{9}$$

$$PLFDM : PLC \times POPC \mapsto \{0, 1\} \tag{10}$$

$$PLFDM\,(PLC_i, POPC_j) = \begin{cases} 1, & \text{if } POPC_j \textbf{ can} \text{ be picked after } PLC_i \\ 0, & \text{if } POPC_j \textbf{ can't} \text{ be picked after } PLC_i \end{cases} \tag{11}$$

The PLFDM values are 0 or 1, depending on pallet loading possibilities. If it is possible to pick the examined product (one element of POPC) after the already picked units (PLC element), then the PLFDM value is 1 (true). Otherwise picking is forbidden, so the PLFDM value is 0 (false) (Eq. (11)). Table 2 and Table 3 illustrate an example PLFDM [6].

Table 2: Pallet Loading Feature based Decision Matrix (PLFDM).

	$POPC_1$	$POPC_2$	$POPC_3$	$POPC_4$	$POPC_5$
PLC_1	1	1	1	1	1
PLC_2	0	1	1	1	1
PLC_3	0	0	1	1	1
PLC_4	0	0	0	1	1
PLC_5	0	0	0	0	1
PLC_6	0	0	0	0	1

Table 3: Pallet Loading Feature based Decision Matrix (PLFDM) example.

	$PlasticBin$	$CartonBox_{HQ}$	$CartonBox_{LQ}$	$SmallBox$	$Fragile$
$PlasticBin$	1	1	1	1	1
$CartonBox_{HQ}$	0	1	1	1	1
$CartonBox_{LQ}$	0	0	1	1	1
$SmallBox$	0	0	0	1	1
$SmallBoxSmallBox$	0	0	0	0	1
$Fragile$	0	0	0	0	1

3.1.2.5 Merging the compatible records in the PLFDM

The resulted PLFDM usually contains records, which have exactly the same values. These PLCs have different attributes but they behave in the same way during order picking. This is the reason why those records can be merged, which can simplify the PLFDM. For example in Table 4, where PLC_5 and PLC_6 are merged this results in $PLC_{5,6}$.

Table 4: Merged Pallet Loading Feature based Decision Matrix (PLFDM).

	$POPC_1$	$POPC_2$	$POPC_3$	$POPC_4$	$POPC_5$
PLC_1	1	1	1	1	1
PLC_2	0	1	1	1	1
PLC_3	0	0	1	1	1
PLC_4	0	0	0	1	1
$PLC_{5,6}$	0	0	0	0	1

3.1.3 Defining the necessity of OPRP-PLF

PLFs are unique characteristics of each warehouse. It is an important factor mainly at distribution warehouses where order picking has a high importance, and the handled products have a huge number of variants. The unique nature of warehousing systems requires that methodology be defined for determining warehouse by warehouse the relevance of OPRP-PLF and the importance of applying optimisation algorithms for it. My methods define the relevance of OPRP-PLF with the evaluation of the modelled pallet loading sequencing possibilities (PLFDM) and with time measurements of warehousing processes. The possible complex PLF based algorithms should be implemented for a warehousing system only when OPRP-PLF is relevant.

3.1.3.1 Pallet Loading Rate (PLR)

After defining the PLFDM of a warehouse, it is possible to evaluate the matrix and define the relevance of PLF based order picking routing optimisation. The Pallet Loading Rate (PLR) is defined based on the amount of pallet loading restrictions; otherwise it is based on the amount of the edges in the PLFDM. Basically, when every POPC can be picked after every PLC, the PLFDM contains 1 in every cell, then the PLF is not relevant at the given warehouse and the PLR is 0. When more and more restrictions (0 value) are defined in the PLFDM then the complexity of OPP and the importance of PLF based routing optimisation is growing. The PLR is calculated by Equation (12), where $MaxNum_e$ is equal to the number of true values in the PLFDM when every POPC element can be picked after every PLC element. Num_e equals to the number of true values in the PLFDM when PLF is modelled [6].

$$PLR = 1 - \frac{Num_e}{MaxNum_e} \tag{12}$$

As part of the research the PLR values have been classified based on intervals, which describe the importance of PLF based routing optimisation at the examined warehouse [6].

- PLF is not relevant, when $PLR = 0$
- PLF is weakly relevant, when $0 < PLR \leqslant 0,2$
- PLF is relevant, when $0,2 < PLR \leqslant 0,4$
- PLF is strongly relevant, when $0,4 < PLR$

Tables 5–8 show examples for each PLR category.

3.1.3.2 Monitoring the OPRP-PLF necessity based on measurements

Measuring the warehousing processes is essential to understand the real nature of the developed warehouse and collect information about the most time consuming movements, relation of causes and effects. The warehousing processes are separable for elementary movements (i.e., travel, administration, pick, search, setup), which can highlight the relevance of PLF at a given warehouse. It is necessary to examine the processes step by step to overlook the sequence, the frequency, the time distribution, and the casual relations of elementary movements. The PLF dependent movements and those that are relevant are different warehouse to warehouse. Some typical steps are the UL reconstruction, travel time, and wrapping [6].

The picker spends time on UL reconstruction when rebuilding the UL structure during order picking. Frequent UL reconstruction movement highlights the importance of PLF and the necessity of SLA re-engineering [6].

When the PLF is relevant at the measured warehouse and re-engineering of the OPP is necessary, the pickers usually move longer distances and have longer routes for similar picking lists and lower the travelling speed during picking to avoid the products from falling down [6].

Wrapping is sometimes necessary to strengthen the picked ULs during the long and complex picking tasks. The wrapped ULs are much more stable, which results in less product damage and higher travel speed during order picking. The frequency and the time requirements of wrapping during OPP are measurable. The necessity of wrapping during picking can highlight the relevance of PLF [6].

Table 5: PLFDM example, when PLF is not relevant

	$POPC_1$	$POPC_2$	$POPC_3$	$POPC_4$	$POPC_5$
PLC_1	1	1	1	1	1
PLC_2	1	1	1	1	1
PLC_3	1	1	1	1	1
PLC_4	1	1	1	1	1
PLC_5	1	1	1	1	1

Table 6: PLFDM example, when PLF is weakly relevant

	$POPC_1$	$POPC_2$	$POPC_3$	$POPC_4$	$POPC_5$
PLC_1	1	1	1	1	1
PLC_2	1	1	1	1	1
PLC_3	1	1	1	1	1
PLC_4	0	0	1	1	1
PLC_5	0	0	0	1	1

Table 7: PLFDM example, when PLF is relevant

	$POPC_1$	$POPC_2$	$POPC_3$	$POPC_4$	$POPC_5$
PLC_1	1	1	1	1	1
PLC_2	1	1	1	1	1
PLC_3	0	0	1	1	1
PLC_4	0	0	0	1	1
PLC_5	0	0	0	0	1

Table 8: PLFDM example, when PLF is strongly relevant

	$POPC_1$	$POPC_2$	$POPC_3$	$POPC_4$	$POPC_5$
PLC_1	0	0	1	1	1
PLC_2	0	0	1	1	1
PLC_3	0	0	0	1	1
PLC_4	0	0	0	0	1
PLC_5	0	0	0	0	0

3.1.4 Summarising statements

Although the order picking routing and the Pallet Loading Problem are important and usually discussed research fields, my state of the art research highlighted that there is lack of harmonising these fields. The specification of the novel Order Picking Routing Problem based on Pallet Loading Feature (OPRP-PLF) resulted in further solutions for modelling and examining the relevance of OPRP-PLF. The warehouses usually lack of appropriate and up-to-date product attributes (dimensions, weight, stacking attribute, etc.) and well-defined stacking constraints.

Thesis statement 1. *I highlighted that there is a relation and a potential synergy between the order picking routing problem and technological issues like packaging and protection of items. Harmonisation of the order picking routing and the Pallet Loading Problem could be necessary to support the picker in stable transport unit building and avoiding product damages during order picking and transport. I proposed the Pallet Loading Feature and the Order Picking Routing Problem based on Pallet Loading Feature (OPRP-PLF). I defined the Pallet Loading Feature as a logistics system attribute, which depends not only on the product attributes but also on the order picking list characteristics and the order picking system itself.*

Thesis statement 1/a. *I defined a methodology for classifying the order picking lines and formalising the pallet loading rules based on known, easily measurable, and rarely changing information. I defined the Pallet Loading Feature based Decision Matrix for formalising the logical picking possibilities of classes.*

Thesis statement 1/b. *Since OPRP-PLF is not relevant for every warehouse, I developed methodologies for defining the relevance of applying OPRP-PLF algorithms, which can be used to examine various warehouses. I defined the Pallet Loading Rate based on the evaluation of the Pallet Loading Feature based Decision Matrix and I highlighted the necessity of the order picking process monitoring.*

My publications related to the statement: [10], [6], [11].

3.2 Complexity of the OPRP-PLF

While examining the complexity of the problems is necessary before developing optimisation algorithms in order to find the right methodology, this section proves the complexity of several OPRP-PLF cases.

My general questions during the examination were as follows:

- How many different picking sequences of the ordered products are possible?
- Is the picker able to find the right sequence of picking herself/himself or is an algorithm necessary?
- What kind of optimisation algorithm is necessary for this kind of problem depending on its complexity?

I defined four industrially relevant cases, when examining the complexity of OPRP-PLF is necessary.

- UL reconstruction is neglected
  - Picking an order, when PLC based **separated zones** are available
  - Picking an order, which can be picked into **1 UL** and order separation is not necessary
  - Picking an order, which should be picked into **more than 1 UL** and order separation is necessary
- Picking an order, when **UL reconstruction** is allowed

I determined the formula for the cases to calculate the possible number of order picking sequencing variations and I examined the behaviour of those in the case of several order picking lists whose length and contents are different. The defined formulas are generally applied standard triangle PLFDM, when $PLC = POPC$ and $SPOPC$ is ignored. This generalisation won't make any significant change in the complexity examination, but it makes a necessary simplification for the transparent calculations.

3.2.1 Complexity of the OPRP-PLF when PLC based separated zones are available

In this case the examined warehouse can store the products in PLC based separated zones. It is a simple case where the warehouse has PLC based zones and the products behave on the same way within the zone. The picker should visit the order picking zones in the right sequence, based on the PLFDM. Within the zone the picking sequence can be simple, based on any dedicated heuristics.

Each order picking list can contain every PLC every time. While the rules of PLFDM are true, any sequence of the PLCs is possible. Therefore the possible cases of my problem is definable with the combination with repetition formula. Equation (13) shows the general formula of the combination with repetition, which results in the number of possible combinations [12].

$$C_k^n = \binom{n+k-1}{k} \tag{13}$$

In this case n means the number of Pallet Loading Classes and k means the length of the order picking list. While the warehouses handle from 1 ($r = 1$) to k ($r = k$) record long order picking lists, we have to summarise the possible combinations from 1 to k (Eq. (14)).

$$\sum_{r=1}^{k} \binom{n+r-1}{r} \tag{14}$$

For example the order picking lists contain maximum 4 records ($k = 4$) and the number of defined PLCs are 4 ($n = 4$). When the length of order picking list equals 1 ($r = 1$), then the number of possible combinations equals n (in my example it equals 4). When the order picking list is 2 records long ($r = 2$), then the number of possible combinations are $\binom{n+1}{2}$. I continue this sequence until $r = k$ and summarize the number of possible combinations to get the dimensions of the optimisation.

Equation (14) is reducible with the telescopic sum terms as shown in Eq. (15).

$$\sum_{r=1}^{k} \binom{n+r-1}{r} = \sum_{r=1}^{k} \left(\binom{n+r}{r} - \binom{n+r-1}{r-1} \right) =$$

$$\left(\binom{n+1}{1} - \binom{n+1-1}{1-1} \right) + \left(\binom{n+2}{2} - \binom{n+2-1}{2-1} \right) +$$

$$\left(\binom{n+3}{3} - \binom{n+3-1}{3-1} \right) + ... +$$

$$\left(\binom{n+k-1}{k-1} - \binom{n+k-1-1}{k-1-1} \right) + \left(\binom{n+k}{k} - \binom{n+k-1}{k-1} \right) =$$

$$\left(\binom{n+1}{1} - \binom{n}{0} \right) + \left(\binom{n+2}{2} - \binom{n+1}{1} \right) + \left(\binom{n+3}{3} - \binom{n+2}{2} \right) + ... +$$

$$\left(\binom{n+k-1}{k-1} - \binom{n+k-2}{k-2}\right) + \left(\binom{n+k}{k} - \binom{n+k-1}{k-1}\right) =$$

$$\left(\binom{n+k}{k} - \binom{n}{0}\right) = \binom{n+k}{k} - 1 \tag{15}$$

The $\binom{n+k}{k} - 1$ formula takes into consideration the PLF rules and defines the possible number of order picking zone sequencing combinations for order picking lists in the case of standard triangle and symmetrical PLFDM. This formula has some important facts:

- Many industrial cases are reducible to standard triangle and symmetrical PLFDM. However some special industrial cases can result in non-symmetrical non-standard triangle PLFDM, which might modify the real number of combinations.
- If I examine the inverse PLFDM, the number of possible combinations will be the same. It highlights, that this formula is not applicable into PLF based routing optimisation without the PLFDM.
- $\binom{n+k}{k} - 1 = \binom{n+k}{n} - 1$, because $\dfrac{(n+k)!}{n! \cdot k!} - 1 = \dfrac{(n+k)!}{k! \cdot n!} - 1$

The goal of the formula is to highlight the complexity of OPRP-PLF, in this case the specific cases and the inverse solutions are negligible. I can conclude, when I implement the PLFs into the order picking routing optimisation, the algorithm should be able to handle unique (non-symmetrical and non-standard triangle) PLFDM with the exact picking rules.

When I will implement a PLF based routing algorithm, the number of PLC-es (n) will be parameter as an output of preceding system monitoring. The algorithm will optimise the route of several order picking lists, whose length (k) will be changing, depending on the customer orders. I examine the exponential growth of combinations with the following limits, where n is an optional high integer number and k goes to infinity. I examine the formula $\binom{n+k}{k}$ (where the -1 is a negligible term in the case of limit value calculation) compared to the e^k formula. Equations (16) and (17) highlight my hypothesis:

- If the defined limit goes to 0 then the formula's growing is faster than e^k (Eq. (16)).
- If the defined limit goes to 1, then the formula's growing is the same as e^k (Eq. (17)).
- If the defined limit goes to ∞, the formula's growing is slower than e^k. In this case, the described specific problem might not require any meta-heuristics optimisation method (Eq. (18)).

$$\lim_{\substack{n \to N \\ k \to \infty}} \frac{e^k}{\binom{n+k}{n}} \overset{?}{=} 0 \tag{16}$$

$$\lim_{\substack{n \to N \\ k \to \infty}} \frac{e^k}{\binom{n+k}{n}} \stackrel{?}{=} 1 \tag{17}$$

$$\lim_{\substack{n \to N \\ k \to \infty}} \frac{e^k}{\binom{n+k}{n}} \stackrel{?}{=} \infty \tag{18}$$

$$\lim_{\substack{n \to N \\ k \to \infty}} \frac{e^k}{\binom{N+k}{N}} = \lim_{\substack{n \to N \\ k \to \infty}} \frac{e^k}{\frac{(N+k)!}{N! \cdot k!}} = \lim_{\substack{n \to N \\ k \to \infty}} \frac{e^k \cdot N! \cdot k!}{(N+K)!} =$$

$$\lim_{\substack{n \to N \\ k \to \infty}} \frac{e^k \cdot \mathbf{N!} \cdot k!}{(N+k) \cdot (N+k-1) \cdot (N+k-2) \cdot \ldots \cdot (N+1) \cdot \mathbf{N} \cdot \ldots \cdot \mathbf{3} \cdot \mathbf{2} \cdot \mathbf{1}} =$$

$$\lim_{\substack{n \to N \\ k \to \infty}} \frac{ke \cdot ((k-1) \cdot e) \cdot ((k-2) \cdot e) \cdot ... \cdot 3e \cdot 2e \cdot 1e}{(N+k) \cdot (N+k-1) \cdot (N+k-2) \cdot ... \cdot (N+3) \cdot (N+2) \cdot (N+1)} = \infty \tag{19}$$

Equation (19) proves, that the formula goes to ∞, because each $\frac{ke}{(N+k)}$, $\frac{((k-1) \cdot e)}{(N+(k-1))}$, $\frac{((k-2) \cdot e)}{(N+(k-2))}$, ... quotient goes to e and there are k quotients, so the result is ∞.

Since $\binom{n+k}{k} - 1 = \binom{n+k}{n} - 1$, we will get the same result if n goes to infinity and k is an optional high integer number.

The result means, that $\binom{n+k}{k} - 1$ has less than exponential growth. Thus, meta-heuristics optimisation method is not necessary, when PLC based separated zones are available.

3.2.2 Complexity of the OPRP-PLF of one UL without order separation

The second case is, when order picking of one UL should be optimised and order separation is not necessary. In this case the customer purchases an order, which will be picked by a picker to one UL. Each order picking list can contain every PLC every time. While the rules of PLFDM are true, any sequence of the PLCs is possible. The main parameters, which influence the number of sequencing variations of a picking list, are the number of records (k), the number of PLCs (n) and the occurrence of the PLCs (i) in the order picking list.

The PLC occurrence is necessary because every PLC contains several products, which usually have their own picking positions. The possible variations of the picking positions within a PLC have to be considered. i is defined from the order picking list point of view, to count the occurrence of PLCs, which are on the order picking list (Eq. (20)). When a PLC occurs i times in a picking list then its possible sequencing variations have to be counted due to the different picking positions ($i!$) (Eq. (21)). The sum of occurrence values (i) has to be equal to the number of order picking list records (k) (Eq. (22)). The number of records (k) has to be equal to or higher than the number of PLCs (n) (Eq. (23)). When $k = n$ then the picking list contains 1 product from each PLC and then the variations of occurrence ($i!$) is not necessary (Eq. (24)).

$$i > 0 \tag{20}$$

$$V = i_1! \cdot i_2! \ldots \cdot i_n! \tag{21}$$

$$k = i_1 + i_2 + \ldots i_n \tag{22}$$

$$k \geq n \tag{23}$$

$$V = i_1 \cdot i_2 \ldots \cdot i_n \tag{24}$$

For example, when the following inputs are given:

- Number of different PLCs on the list equals 3 ($n = 3$), $PLC = \{A, B, C\}$
- Number of order picking list records equals 12 ($k = 12$)
- A PLC occurs 5 times, $i_1 = 5$
- B PLC occurs 4 times, $i_2 = 4$
- C PLC occurs 3 times, $i_3 = 3$

The number of variations for the above mentioned example is $5! \cdot 4! \cdot 3! = 17280$.

This example is just one case, each n and k pairs have several combinations depending on the occurrence of PLCs. Equation (25) shows the formula, which defines the number of possible combinations.

$$C_{k;n} = \binom{k-1}{n-1} \tag{25}$$

One possible combination is when the occurrence values are balanced ($i_1 \approxeq i_2 \approxeq \ldots \approxeq i_n$) and this reaches the minimum number of variations. In the case of the mentioned example the minimum number of variations equals 13824 when:

- $n = 3$
- $k = 12$
- A PLC occurs 4 times
- B PLC occurs 4 times

- C PLC occurs 4 times

Another possible combination is when one of the occurrence values is maximum and the others equal 1, and this reaches the maximum number of variations (Eq. (26)).

$$(k - (n - 1))! \cdot (1!)^{n-1} = (k - n + 1)! \tag{26}$$

In the case of the mentioned example the maximum number of variations equals 3628800, when:

- $n = 3$
- $k = 12$
- A PLC occurs 10 times
- B PLC occurs 1 times
- C PLC occurs 1 times

During the complexity examination the maximum formula is going to be used (Eq. (26)) because the order picking routing optimisation algorithm has to be able to handle this case as well, which results in the highest number of variations and causes the highest complexity.

The $(k - n + 1)!$ formula (Eq. (26)) takes into consideration the PLF rules in the case of standard triangular and symmetrical PLFDM and it has some important facts:

- Many industrial cases are reducible to standard triangular and symmetrical PLFDM. However, some special industrial cases can result in neither symmetrical nor standard triangular PLFDM, which might modify the real number of variations.
- When the inverse PLFDM is examined, the number of possible variations will be the same. It highlights, that the proposed formula is not applicable in PLF based order picking routing optimisation without the PLFDM.
- The proposed formula assumes that each product has only 1 picking position. However, sometimes more than 1 picking position of a product is also possible.

The aim of this formula (Eq. (26)) is to highlight the importance and complexity of PLF based order picking routing optimisation. In this case the specific cases and the inverse solutions are negligible. It could be said that when PLFs are implemented into the order picking routing optimisation, the algorithm should be able to handle a unique (non-symmetrical and non-standard triangular) PLFDM with the exact picking rules.

Generally, the necessary data (n, k, i_1, i_2,… i_n) regarding the order picking list will be available during PLF based order picking routing optimisation to see each possible variation. The algorithm will optimise the picking sequence of several order picking lists whose parameters will be different depending on the customer orders.

The $(k-n+1)!$ formula (Eq. (26)) is compared to the e^k formula to examine the complexity of the OPRP-PLF. Equation (27) can prove the exponential growth of variations, where n is an optional constant and k goes to infinity.

$$\lim_{k\to\infty} \frac{e^k}{(k-n+1)!} = \\ \lim_{k\to\infty} \frac{e \cdot e \cdot e \cdot \ldots \cdot e}{(k-n+1)\cdot(k-n)\cdot(k-n-1)\cdot\ldots\cdot 3\cdot 2\cdot 1\cdot\ldots\cdot 1} = 0 \tag{27}$$

Equation (27) goes to 0 because each $\frac{e}{(k-n+1)}, \frac{e}{(k-n)}, \frac{e}{(k-n-1)} \ldots$ quotient goes to 0 and the further quotients $\left(\frac{e}{2}, \frac{e}{1} \ldots\right)$ are constants. This result means that the number of variations ($(k-n+1)!$) has a stronger growth than the e^k has. It proves that the proposed formula has at least exponential growth. It could be said that a meta-heuristics optimisation method would be necessary for PLF based routing optimisation.

Each warehouse handles shorter and longer picking lists. Table 9 represents an example when $n = 3$ and k is between 1 and 15. In this case when $k = 12$ and $n = 3$, which is definitely possible in real life, the possible sequencing variations equal 3628800. It could be said that it is impossible for the picker to be able to define a nearly optimal sequence by herself/himself without any support. Naturally, there are possible cases when k and n are smaller but in this case complex meta-heuristics optimisation might not be relevant (i.e., in Table 9, when $n = 3$ and $k = 6$ there are 24 possible variations). It is necessary to examine the nature of picking lists on a tactical level and determine whether the order picking lists require complex PLF based optimisation or not. When there is a possibility for several complex order picking lists then implementation of PLF based meta-heuristics routing optimisation is necessary. However, the Warehouse Management System should be able to decide on an operational level which list will be sequenced by complex and maybe time consuming algorithms and which will be handled by simple algorithms or by the picker herself/himself.

3.2.3 Complexity of the OPRP-PLF when order is separated to several ULs

This section examines the case when the purchased order should be separated to ULs because the ordered amount of products is higher than 1 UL's capacity. The defined ULs have the same parameters and behave in the same way as the previously discussed picking

Table 9: Number of order picking sequencing variations, when $n = 3$ and k is growing.

k	MaxVar	e^k
1	0	2.72
2	0	7.39
3	1	20.09
4	2	54.60
5	6	148.41
6	24	403.43
7	120	1 096.63
8	720	2 980.96
9	5 040	8 103.08
10	40 320	22 026.47
11	362 880	59 874.14
12	3 628 800	162 754.79
13	39 916 800	442 413.39
14	479 001 600	1 202 604.28
15	6 227 020 800	3 269 017.37

lists. My questions are extended as follows:

- How many different variations are possible for separating an order and sequencing the picking products of each UL?
- Is the picker able to find the right separation of an order and picking sequence of each UL herself/himself or is an algorithm necessary?
- What kind of optimisation algorithm is necessary for these kinds of problems depending on its complexity?

The main parameters, which influence the number of separating and sequencing variations of an order, are the number of records (Order:K, UL:k), the number of PLCs (Order:N, UL:n) and the occurrence of the PLCs (Order:I, UL:i).

It is assumed that the number of possible ULs could be equal to or lower than the number of ordered records (K) (Eq. (28)). The sum of each UL's length (k_1, $k_2 \ldots k_{UL_{num}}$) equals K(Eq. (29)). The order picking sequencing variations of each possible UL are counted by the previously defined formula, $V = i_1! \cdot i_2! \ldots \cdot i_n!$ (Eq. (21)). Based on the combination with repetition formula, Eq. (30) defines the possible order separation combinations, where $UL_{num} = K$, because of Eq. (28).

$$UL_{num} \leqslant K \tag{28}$$

$$K = k_1 + k_2 + \ldots + k_{UL_{num}} \tag{29}$$

$$\binom{UL_{num}+K-1}{UL_{num}-1} = \binom{2 \cdot K - 1}{K} \tag{30}$$

Equation (31) sums up the possible separating combinations and sequencing variations of each UL.

$$\sum_{k_1,k_2,\ldots,k_K=0}^{k_1+k_2+\ldots+k_K=K} \binom{K}{k_1} \cdot V_{k_1} + \binom{K-k_1}{k_2} \cdot V_{k_2} + \ldots + \binom{K-k_1-k_2-\ldots-k_{K-3}-k_{K-2}}{k_{K-1}} \cdot V_{k_{K-1}} + \binom{K-k_1-k_2-\ldots-k_{K-2}-k_{K-1}}{k_K} \cdot V_{k_K} \tag{31}$$

The model counts using the maximum number of variations of each UL, when $i >= 0$, thus $V_{k_j} = k_j!$. Differently from Eq. (20), $i = 0$ is allowed because in this case the order picking lists are combined during examination, thus the exact occurrence of each n is unknown. It is a specific case, which results in the highest number of variations, although during optimisation the exact occurrence of each n might be known. In this case the proposed formula can be simplified as Eq. (32) shows. Some further simplifications result in Eq. (33), which defines the possible variations for separating an order and sequencing the picking products of each UL.

$$\sum_{k_1,k_2,\ldots,k_K=0}^{k_1+k_2+\ldots+k_K=K} \frac{K!}{(K-k_1)!} + \frac{(K-k_1)!}{(K-k_1-k_2)!} + \ldots + \frac{(K-k_1-k_2\ldots k_{K-2})!}{(K-k_1-k_2\ldots k_{K-1})!} + \frac{(K-k_1-k_2\ldots k_{K-1})!}{(K-k_1-k_2\ldots k_K)!} \tag{32}$$

$$\sum_{k_1,k_2,\ldots,k_K=0}^{k_1+k_2+\ldots+k_K=K} \frac{K!}{(K-k_1)!} + \frac{(K-k_1)!}{(K-k_1-k_2)!} + \ldots + \frac{(k_{K-1}+k_K)!}{k_K!} + \frac{k_K!}{0!} \tag{33}$$

Equation (27) proves, that sequencing 1 UL is at least an exponential problem and requires meta-heuristics optimisation. When an algorithm separates orders to ULs and sequences each UL it will be at least an exponential problem as well, which requires meta-heuristics optimisation. It could be said that supporting the picker with a separating and sequencing algorithm is even more important in this case because this problem is even more complex.

3.2.4 Complexity of the OPRP-PLF, when UL reconstruction is allowed

When reconstruction is allowed, then the possible sequencing combinations equal $k!$, where k is the number of the records on the order picking list. This case results in more possible combinations than the previously described case when the order separation in not necessary ($k! \geq (k - n + 1)!$). When the number of PLCs equals one ($n = 1$), then the two cases are the same. Furthermore, while Equation 34 goes to 0, $k!$ goes to infinity quicker then e^k.

$$\lim_{k \to \infty} \frac{e^k}{k!}{=}0 \tag{34}$$

These facts highlight, that this case is more complex than the previous ones. Thus the exponential complexity and the short available running time make the application of meta-heuristics algorithm necessary. I proposed, that evolutionary optimisation could be one of the suitable meta-heuristics methodology for the problem.

3.2.5 Summarising statements

While the order picking routing algorithms do not consider the Pallet Loading Features, development of algorithms would be necessary for the OPRP-PLF. Algorithm development for novel problems generally should be started with complexity evaluation of relevant industrial cases to define the proper and necessary methodology for optimisation.

Thesis statement 2. *My examination highlighted that meta-heuristics optimisation method is not necessary for OPRP-PLF, when Pallet Loading Class based separated zones are available. However, I proved, that meta-heuristics optimisation method is necessary in the following cases because of the at least exponential growth of the possible order picking sequencing combinations and the low available running time, since the order picking zone is not separated based on Pallet Loading Classes.*

- *Order picking of one unit load without order separation, when unit load reconstruction is not allowed during the order picking.*
- *The order is separated into several unit loads, and unit load reconstruction is not allowed during the order picking.*
- *Unit load reconstruction is allowed during the order picking of one unit load.*

While the picker should get the optimised picking list without wasting time in the daily operation, the algorithm running time is a critical factor. Relying upon these facts, I proposed, that evolutionary optimisation would be one of the suitable meta-heuristics methodologies for the problem.

My publications related to the statement: [7], [11].

3.3 Algorithms for OPRP-PLF

The complexity of OPRP-PLF has been examined in the previous section (Section 3.2.). It resulted in a number of possible sequencing combinations having exponential growth when the length of the picking lists goes to infinity. It could be said that the suitable order picking sequence has to be available to the picker quickly after receiving the order due to its operational nature. The possibly high number of combinations and the low available time make a meta-heuristics optimisation method necessary for OPRP-PLF to support the pickers with an effective picking sequence. It is necessary to examine the nature of picking lists on a tactical level warehouse by warehouse and determine whether the characteristics of order picking lists require complex PLF based optimisation or not. When there is a possibility for several complex order picking lists with a high number of possible sequencing combinations then implementation of PLF based meta-heuristics routing optimisation is necessary. In my research I focus on evolutionary algorithms as a possible meta-heuristics approach [11].

Naturally, there are possible cases for simple and short picking lists or PLF based separated picking zones when a strict PLF based picking sequence is definable. In this case complex meta-heuristics optimisation might not be relevant, proper SLA and simple heuristics (e.g., s-shape, largest gap, etc. [57]) usually can define the right picking sequence [7].

First, this section introduces the objective function defining method for evaluating order picking sequence in the case of relevant PLF. I made an analytical examination for simple cases of the OPRP-PLF to highlight the operative decisions of the pickers. I defined Bacterial Memetic Algorithm (BMA) and Simulated Annealing (SA) algorithms for the OPRP-PLF, when PLF is relevant and the order picking is operating in one zone for one UL without order separation. I applied the algorithms for both cases when reconstruction is allowed (Section 3.2.4.) and when it is not allowed (Section 3.2.2.).

3.3.1 Objective function for PLF based evaluation of order picking sequence

I evaluate the possible order picking sequencing solutions based on time requirements. Counting the lead time of each picking task begins when the picker starts the list and picks up an empty pallet at the DA position. The lead time measurement is finished when the picker has transported the ready UL to the DA position. During order picking the picker visits each picking position of list, picks the ordered products and reconstructs the UL structure, if it is necessary. The lead time (T) is the sum of the travel time (T_T), the picking time (T_P), the reconstruction time (T_R), and the other times (T_O) (Eq. (35)). α, β, γ, and δ are weighted constraints.

$$T = \alpha \cdot T_T + \beta \cdot T_P + \gamma \cdot T_R + \delta \cdot T_O \tag{35}$$

$$\begin{gathered} min\,(\alpha \cdot T_T + \beta \cdot T_P + \gamma \cdot T_R + \delta \cdot T_O) \\ min\,(T) \end{gathered} \tag{36}$$

The travel time is defined based on the summarised travelled distance and the speed of the picker (v) (Eq. (37)). The travel time is measured when the picker starts from the departure position, moves between picking positions and arrives to the destination position of the UL. S_{r_x,r_y} is the distance from the xth record to the yth record. The DA is the departure and arrival position, where the picker starts and finishes the picking task. DA equals r_0.

$$T_T = \frac{S_{r_k,DA}}{v} + \sum_{i=1}^{k} \frac{S_{r_{i-1},r_i}}{v} \tag{37}$$

The picking time (T_P) is the sum of the time requirements of the labour intensive picking job when the picker moves the products from the picking position to the pallet (Eq. (38)). t_P is the time required to pick 1 record on the picking list. T_P depends on several factors, for example, the ordered quantity, weight, shape and packaging solution of the ordered product, but at this stage of this research the simple constraint (t_P) based handling is sufficient to demonstrate the effectiveness of the model. The picking time definition module is upgradable later warehouse by warehouse.

$$T_P = \sum_{i=1}^{k} t_{P_i} \tag{38}$$

The proposed model allows the reconstruction of UL during order picking. The reconstruction time (T_R) defines the time needs of the movements during the picking process when the next products are not allowed to be picked after the previously picked products because of stability reasons. Then, unit load reconstruction is necessary. The picker should take the products from the picked unit load off while the next product on the list can be picked. The picker then builds back the unit load with products which sat apart. The model assumes that the picker separates the unpicked products around the unit load and moves those back in the right PLFDM based sequence. It is definitely a simplified tower of Hanoi problem, where an infinite number of towers is possible. Equation (39) defines the reconstruction time (T_R) requirements of a list, where r_p is the number of problematic records which need reconstruction. r_r is the number of records with a stronger PLC after the problematic record, whose products will be taken off and put back. Similar to the picking time, the reconstruction time (t_R) is also a constraint at this stage of the research. This paper proves that reconstruction during order picking can result in a shorter load time in the case of some circumstances.

$$T_R = \sum_{i=0}^{r_p} r_{r_i} \cdot 2 \cdot t_{R_i} \tag{39}$$

T_O defines the extra time needs, for example administration, searching, labelling, etc. It was neglected in this model because it is not influenced directly by the picking sequence.

The OPRP-PLF is a minimisation problem. The overall aim is to minimise the order picking lead time (T) of stable unit loads (Eq. (36)) subject to k (number of records on the order picking list), r_p, r_r, and the distances between positions S.

3.3.2 Analytic examination of simple cases

Analytic examination helps understand the nature of the order picking decisions when PLF is a relevant factor. It highlights the necessity of supporting the pickers in the case of OPRP-PLF.

When the picking lists are short (k is low, like 2-6 records) and/or the list is simple (contains low number of POPC – n is low) then the picker is usually able to define the optimal sequence for picking, which results in the shortest picking lead time. This subsection describes some simple picking lists and their calculated objective functions. The lead times are calculated using the previously described formulas based on a simple distance matrix (Table (10)), where the values are defined in meters. The possible UL building rules are described in Table (11). Obviously in the explained cases the picker defines the right sequence without any calculation, based merely on experience and best practices.

Table 10: Distance matrix for the simple cases.

	Pos 1	Pos 2	Pos 3	Start-End
Pos 1	0	50	20	100
Pos 2	50	0	30	80
Pos 3	20	30	0	10
Start-End	100	80	10	0

Table 11: PLFDM for the simple cases.

	A	B	C
A	1	1	1
B	0	1	1
C	0	0	1

The first simple case has 2 records ($k = 2$) and contains 2 POPCs ($n = 2$). Table (12) shows a possible picking sequence when UL reconstruction is necessary based on the PLFDM because the record number 1 (POPC attribute is "B") is picked before the record number 2 (POPC attribute is "A"). Table (13) describes a better picking sequence when reconstruction is not necessary and the travel time is equal to the previously discussed solution.

The second simple case has 3 records ($k = 3$) and contains 3 POPCs ($n = 3$). Table (14) and Table (15) evaluate 2 possible picking sequences when reconstruction is necessary and when it is not required, respectively. The results show that the second solution's lead time is lower when reconstruction is not necessary. It is highlighted that in this case the picker has to travel a longer route to avoid reconstruction and reach a lower lead time. It could be said that the picker has to take into consideration the PLF and not to minimise the route length. When the number of records is higher and/or the picking list is more complex the picker won't be able to make the right decision without any IT support, which defines the nearly optimal sequence.

Table 12: Simple case 1 ($k = 2$ and $n = 2$) **with** reconstruction.

Record ID	Position	POPC	T_P	r_{r_i}	T_R	S_{r_{i-1},r_i}	T_T	Lead Time
1	Position 2	B	00:10	0	00:00	80	00:50	
2	Position 1	A	00:10	1	00:15	50	00:31	
	Start-End					100	01:02	
Sum			00:20		00:15	230	02:23	**02:58**

Table 13: Simple case 1 ($k = 2$ and $n = 2$) **without** reconstruction.

Record ID	Position	POPC	T_P	r_{r_i}	T_R	S_{r_{i-1},r_i}	T_T	Lead Time
2	Position 1	A	00:10	0	00:00	100	01:02	
1	Position 2	B	00:10	0	00:00	50	00:31	
	Start-End					80	00:50	
Sum			00:20		00:00	230	02:23	**02:43**

Table 14: Simple case 2 ($k = 3$ and $n = 3$) **with** reconstruction.

Record ID	Position	POPC	T_P	r_{r_i}	T_R	S_{r_{i-1},r_i}	T_T	Lead Time
3	Position 3	B	00:10	0	00:00	10	00:06	
1	Position 2	C	00:10	1	00:15	30	00:19	
2	Position 1	A	00:10	2	00:30	50	00:31	
	Start-End					100	01:02	
Sum			00:30		00:45	190	01:58	**03:13**

Table 15: Simple case 2 ($k = 3$ and $n = 3$) **without** reconstruction.

Record ID	Position	POPC	T_P	r_{r_i}	T_R	S_{r_{i-1},r_i}	T_T	Lead Time
2	Position 1	A	00:10	0	00:00	100	01:02	
3	Position 3	B	00:10	0	00:00	20	00:13	
1	Position 2	C	00:10	0	00:00	30	00:19	
	Start-End					80	00:50	
Sum			00:30		00:00	230	02:24	**02:54**

3.3.3 Bacterial Memetic Algorithm for OPRP-PLF

The proposed OPRP-PLF is a combinatorial optimisation problem whose exponential complexity and the short available running time make the application of a meta-heuristics approach such as an evolutionary algorithm necessary. The algorithm should define an approximated optimum within a short time from the huge amount of possible combinations without evaluating each possibility. I offer the Bacterial Memetic Algorithm (BMA) for the defined problem, because it has already been successfully utilised in several combinatorial optimisation problems because of its fast convergence speed (TSP [32], robot path planning [17], feature selection problem [74]); and in continuous optimisation tasks as well (fuzzy rule base extraction [16], building energetics [25], robot locomotion [59]). The applied local search operator helps the algorithm converge to the global optimum. BMA has not yet been used for the order picking routing problem, particularly for the OPRP-PLF. I propose a BMA algorithm for OPRP-PLF to sequence a given order picking list while following the pallet loading rules and to minimise the order picking lead time.

The advantages of BMA for OPRP-PLF are the well scalable nature and the high convergence speed. The disadvantages are the too sophisticated structure, because it requires costly experts for industrial implementation and support; and the parameter setting which needs some expertise. However, this sophisticated structure makes it scalable and quick. A further weakness is, that the human workers might not be able to understand the logic behind the obtained quasi-optimal result, which can frustrate them. The human workers have to rely more on the system. Nevertheless, the low convergence time and the scalable structure compensate the weaknesses. A further benefit of BMA over other nature inspired optimisation methods is it applicability on combinatorial problems. Many evolutionary and swarm based optimisation techniques (such as Particle Swarm optimisation (PSO) [39], Differential Evolution (DE) [65], Artificial Bee Colony algorithm (ABC) [38], and many more) work well on continuous optimisation, however they are not prepared for solving combinatorial optimisation problems.

The BMA algorithm operates in four steps (Alg. 1). The first step is to create an initial population with N_{ind} individuals (bacterium), each bacterium represents a solution to the original problem. This can be done randomly or some further rules can be defined. Next, bacterial mutation (BM) and local search (LS) methods are utilized for the whole population then a gene transfer (GT) operator is executed. These 3 operators are repeated while the stopping condition is achieved. The stopping condition is usually given by a predefined maximum number of generations (N_{gen}) [16, 32].

```
1  Execute initial population generation method
2  for $i := 1$ to $N_{ind}$ do
3      Evaluation of each individual to define objective function $T$
4  end
5  for $g := 1$ to $N_{gen}$ do
6      for $i := 1$ to $N_{ind}$ do
7          Execute Bacterial Mutation operator
8          Execute Local Search operator
9      end
10 end
11 for $i := 1$ to $N_{inf}$ do
12     Order the population in ascending order by the objective function $T$
13     Execute Gene Transfer operator
14 end
```

Algorithm 1: Applied BMA procedure

The further sections present operator alternatives for OPRP-PLF. The combination of these operators will be tested and evaluated.

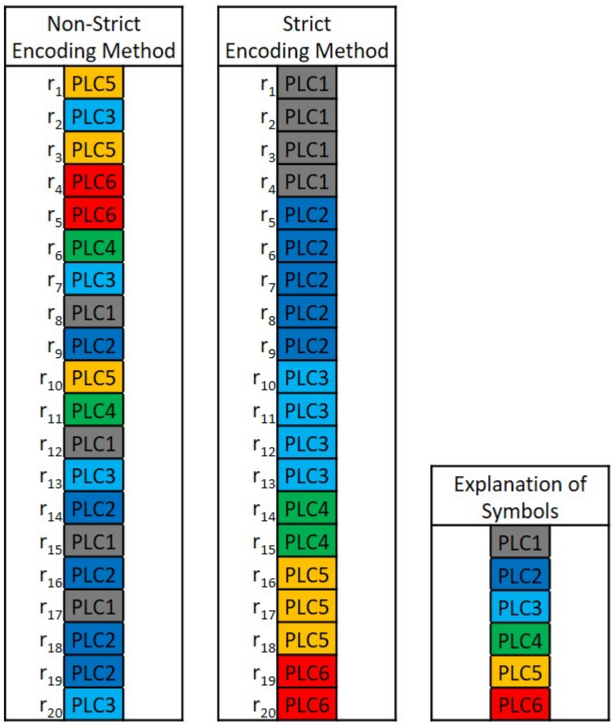
3.3.3.1 Encoding Methods of Individuals

The proposed BMA algorithms are defined for optimising the order picking list of a UL, which contains k different order picking positions without the departure and arrival position. Each bacterium represents a possible picking sequence of the order picking positions. The length of an individual is equal to the length of the picking list (k) because every picking position has to be visited once. Non-strict and strict encoding methods have been applied. The non-strict encoding method randomly sequences each picking position without any rules. The strict encoding method sequences the picking positions in a strict PLC sequence. The first PLC has the highest picking possibility in the PLFDM (e.g, heavy, strong and big packages), and the last PLC has the lowest possibility (e.g., fragile products). The picking position sequence within a PLC is randomised. Figure 8 demonstrates both encoding methods for a 20 record long ($k = 20$) order picking list where the records with the same PLC attribute have the same colour and the sequence gives the picking sequence.

3.3.3.2 Objective Function for Individual Evaluation

The individuals are evaluated several times during BMA optimisation, first, after the initial population generation. The evaluating method calculates the order picking lead

Figure 8: Encoding methods



time of the picking list based on the previously described objective functions (Section 3.3.1).

3.3.3.3 Initial Population Generation

The initial population generator defines N_{ind} number of individuals. These represent the possible picking sequences of the order picking positions. Non-strict and strict initial population generators have been defined based on the non-strict and the strict encoding method.

3.3.3.3.1 Non-strict Initial Population with an Eugenic Bacterium The order picking list has an unstructured sequence, which is the initial state of each individual. Based on the non-strict encoding method (Alg. 2) the initial sequence of the whole list is perturbed N_{ind} times to define the individuals. During the perturbation the algorithm randomly selects a position in the list for each element and changes the given element with the element of the selected position. The random selection means that the algorithm defines a number between 1 and k with equal probability. This perturbation method can be used for separated parts of the whole picking list, which is useful for the perturbation of separated segments.

Eugenic elements can be used in the initial population creation and in bacterial mutation operators [73]. Eugenics can model the pickers' best practices and put some deterministic elements into the algorithm. When the *Eugenic* parameter of the algorithm

```
for $i := 1$ to $N_{ind}$ do
    if $i = 1$ and $Eugenic = true$ then
        Randomised perturbation of every record on the list
        Order the list in ascending order by theoretical PLC sequence
    else
        Randomised perturbation of every record on the list
    end
end
```

Algorithm 2: Non-strict Initial Population with a Eugenic Bacterium Procedure

is true then the first individual is defined based on the strict encoding method (Alg. 3). First, the initial state of the picking list is perturbed randomly as a non-strict individual. Then the list is ordered by the theoretical POPC sequence, which means an ascending order of the POPCs based on the physically possible loading sequence. The heaviest POPC is the first POPC and the most fragile one is the last POPC. This method might help the algorithms with a fairly good approximation.

```
for $i := 1$ to $N_{ind}$ do
    Randomised perturbation of every record on the list
    Order the list in ascending order by theoretical PLC sequence
end
```

Algorithm 3: Strict Initial Population Procedure

3.3.3.3.2 Strict, PLF based Initial Population The strict initial encoding method defines each individual based on the strict encoding method, similar to how the above-mentioned eugenic individual is defined. Its result is that each individual has a strict, PLC based sequence. It models that case when the picker gets a previously sequenced list, which will be fine tuned by him/her.

3.3.3.4 Bacterial Mutation (BM)

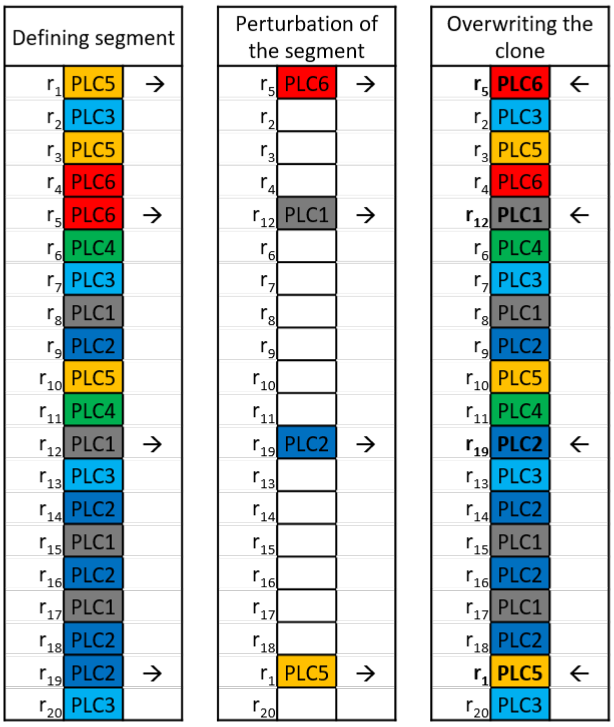
The bacterial mutation (BM) operator mutates the picking position sequence of one bacterium. First, $N_{clones} + 1$ copies (clones) of the bacterium are generated. Then a certain segment of the bacterium is randomly selected and the parameters of the selected segment are randomly changed separately in each clone except one which is left unmutated. This segment mutation is done by the perturbation method, which was used during the initial population definition. Every clone is evaluated, the best clone is selected, and the best mutated segment is transferred into the other clones. This process continues until every segment of the bacterium has been mutated and tested. At the end of this process the best clone overwrites the original bacterium and the clones are eliminated [32].

The developed BM operators differ in segment creation and transferred segment definition methods.

3.3.3.4.1 Non-strict Bacterial Mutation During the segment definition (Alg. 4), each element of the bacterium (records of the order picking list) is allocated to a segment. The algorithm selects a record from the set of unselected records randomly, with equal probability. The selected record is allocated to the current segment, while the number of allocated elements reaches the defined segment length ($L_{segment}$) then new segment is defined. The number of segments depends on the length of order picking list (k) and the segment length parameter ($L_{segment}$). The last segment can be shorter than $L_{segment}$. Then the sequence of the segments is randomly permutated to define the segment order for perturbation.

Figure 9 illustrates the permutation of one clone, one segment in the case of Non-strict bacterial mutation, where $k = 20$ and $L_{segment} = 4$. After perturbing the same segment of each clone, the clones are evaluated and sorted by the lead time. The first clone's segment, which performed with the lowest lead time is transferred to the other clones. The perturbation, the evaluation, and the segment transfer are done segment by segment.

Figure 9: Non-strict bacterial mutation



Defining segment			Perturbation of the segment			Overwriting the clone		
r_1	PLC5	→	r_5	PLC6	→	r_5	**PLC6**	←
r_2	PLC3		r_2			r_2	PLC3	
r_3	PLC5		r_3			r_3	PLC5	
r_4	PLC6		r_4			r_4	PLC6	
r_5	PLC6	→	r_{12}	PLC1	→	r_{12}	**PLC1**	←
r_6	PLC4		r_6			r_6	PLC4	
r_7	PLC3		r_7			r_7	PLC3	
r_8	PLC1		r_8			r_8	PLC1	
r_9	PLC2		r_9			r_9	PLC2	
r_{10}	PLC5		r_{10}			r_{10}	PLC5	
r_{11}	PLC4		r_{11}			r_{11}	PLC4	
r_{12}	PLC1	→	r_{19}	PLC2	→	r_{19}	**PLC2**	←
r_{13}	PLC3		r_{13}			r_{13}	PLC3	
r_{14}	PLC2		r_{14}			r_{14}	PLC2	
r_{15}	PLC1		r_{15}			r_{15}	PLC1	
r_{16}	PLC2		r_{16}			r_{16}	PLC2	
r_{17}	PLC1		r_{17}			r_{17}	PLC1	
r_{18}	PLC2		r_{18}			r_{18}	PLC2	
r_{19}	PLC2	→	r_1	PLC5	→	r_1	**PLC5**	←
r_{20}	PLC3		r_{20}			r_{20}	PLC3	

3.3.3.4.2 Non-strict Bacterial Mutation with Simulated Annealing This operator (Alg. 5) defines the segments like the previous one but the segment overwriting method is completed using the SA method. The assumption is that the SA method could increase the picking position mixing efficiency during bacterial mutation.

```
for $c := 1$ to $N_{clones} + 1$ do
    Define clone $c$ by copying the original individual
end
Declare integer variable $Segment := 1$
Declare integer variable $SegmentLength := 0$
for $i := 1$ to $k$ do
    Randomly allocate one, unselected record to the current segment ($Segment$)
    $SegmentLength$ += 1
    if $SegmentLength = L_{segment}$ then
        Define new segment
        $Segment$ += 1
        $SegmentLength := 0$
    end
end
Perturb the sequence of the segments
for $s := 1$ to $Segment$ do
    for $c := 2$ to $N_{clones} + 1$ do
        Perturb the segment $s$ of the clone $c$
    end
    for $c := 1$ to $N_{clones} + 1$ do
        Evaluate clone $c$
    end
    Find the best clone ($BestClone$) based on $T$
    Copy segment $s$ of $BestClone$ to each clone
end
Overwrite the original individual by the best clone
```

Algorithm 4: Non-strict Bacterial Mutation Procedure

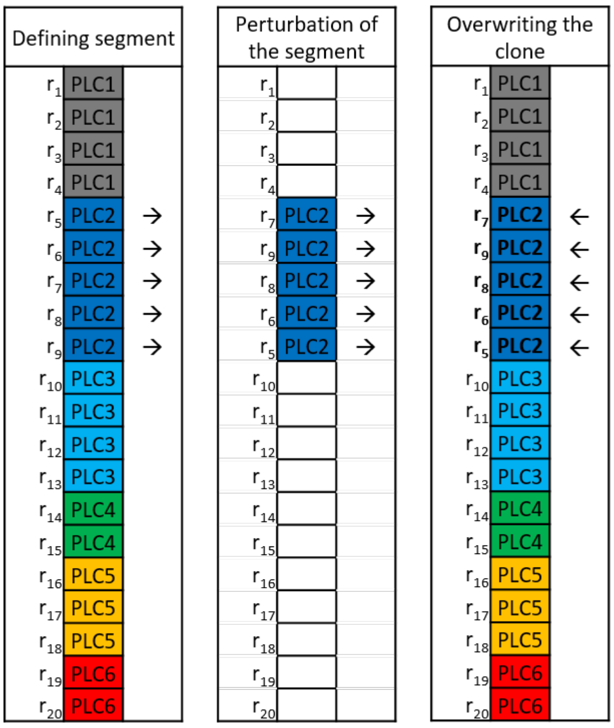
After perturbation and evaluation, the segment for transfer is defined based on Eq. (40), where $OverwritePortion \sim U(0, 100)$ is a uniform random number, gen is the current generation, N_{gen} is the total number of generations, and τ is a parameter. If Eq. (40) holds then the transferred segment is the segment of the second best clone, otherwise the segment is defined based on the best clone.

$$OverwritePortion < e^{-\frac{gen \cdot \tau}{N_{gen}}} \tag{40}$$

```
for $c := 1$ to $N_{clones} + 1$ do
    Define clone $c$ by copying the original individual
end
Declare integer variable $Segment := 1$
Declare integer variable $SegmentLength := 0$
for $i := 1$ to $k$ do
    Randomly allocate one, unselected record to the current segment ($Segment$)
    $SegmentLength$ += 1
    if $SegmentLength = L_{segment}$ then
        Define new segment
        $Segment$ += 1
        $SegmentLength := 0$
    end
end
Perturb the sequence of the segments
for $s := 1$ to $Segment$ do
    for $c := 2$ to $N_{clones} + 1$ do
        Perturb the segment $s$ of the clone $c$
    end
    for $c := 1$ to $N_{clones} + 1$ do
        Evaluate clone $c$
    end
    Order the clones in ascending order by the objective function $T$
    if Equation (40) holds then
        Define BestClone as the 2nd clone
    else
        Define BestClone as the 1st clone
    end
    Copy segment $s$ of BestClone to each clone
end
Overwrite the original individual by the best clone
```
Algorithm 5: Non-strict Bacterial Mutation with Simulated Annealing Procedure

3.3.3.4.3 Strict, PLF based Bacterial Mutation The strict BM operator defines the segments based on the PLC parameter of the order picking records. The picking sequence of PLCs won't change during BM but the records are perturbed within the PLC based segments. It results in different routes with the same pallet loading attribute. Each PLC of the order picking list defines one segment. The number of segments depends on the number of different PLCs in the order picking list. The length of the segments depends on the number of records with the same PLC parameter in the order picking list. The segment transfer is simple, without the SA method. Figure 10 illustrates the permutation of one clone, one PLC homogeneous segment in the case of strict bacterial mutation, where $k = 20$, the number of segments is 6, and $L_{segment}$ of the selected segment is 5.

Figure 10: Strict bacterial mutation



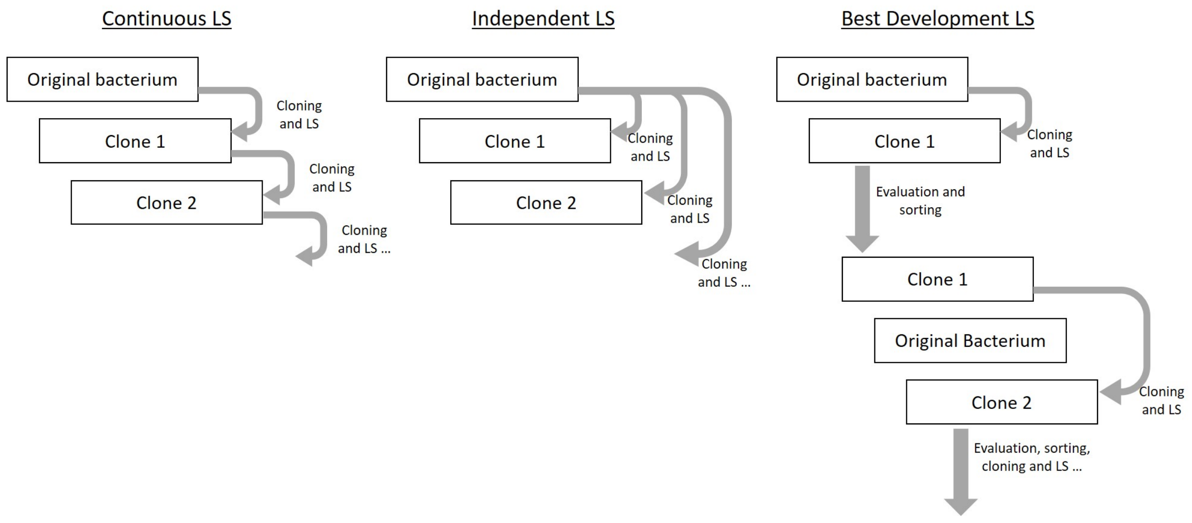
3.3.3.4.4 Strict, PLF based Bacterial Mutation with Simulated Annealing

This operator works with the same segment definition method like the previous one but the segment transfer is completed with the same SA method, which has been mentioned above for the SA completed non-strict BM operator.

3.3.3.5 Local Search (LS)

After the BM operator a problem specific local search is applied, which responds to improve the given bacterium. The LS operator makes a clone and tries to improve the bacterium. It is repeated as many times as defined by the LS_{repeat} parameters. This paper describes 3 LS operators, which have strict and non-strict alternatives. All 6 LS operators have an alternative, which is completed using the SA method. Figure 11 demonstrates the cloning methodology of the 3 main LS operators, which will be discussed in the further subsections.

Figure 11: Cloning methodology of the 3 main LS operators



3.3.3.5.1 Continuous Local Search The "Continuous" LS operator (Alg. 6) copies the bacterium then 2 positions are randomly selected and their picking sequence is defined based on PLFDM. If this sequence is different from their current sequence then the 2 positions are exchanged. This local search algorithm is repeated LS_{repeat} times by copying the previously defined LS clone of this bacterium. When the clones are done each of them are evaluated and sorted by the lead time. If the first clone achieved a lower lead time than the original one then the original bacterium is overwritten. Figure 12 depicts a Non-strict LS operator of one clone, where $k = 20$.

3.3.3.5.2 Independent Local Search The "Independent" LS operator (Alg. 7) applies the previously described position changing method for the original bacterium LS_{repeat} times, independently from the previous clone of this bacterium. The evaluating and overwriting procedures are the same like in the case of "Continuous" LS.

3.3.3.5.3 Best Development Local Search The Best Development LS operator (Alg. 8) evaluates and sorts the clones after each clone definition and possible modification. The LS method runs for the current best clone LS_{repeat} times.

3.3.3.5.4 Strict Local Search Alternatives Each previously described LS operator has a strict alternative, which is used when a strict initial population and a strict BM are used. First, a segment is defined randomly, which contains elements with the same PLC attribute. Two random positions are selected from this segment and those are always

```
Copy the original individual (It will be the first individual (individual 0) for LS)
for $c := 1$ to $LS_{repeat}$ do
    Clone the previous $(c-1)$ individual
    Declare integer variable $rnd1$
    Declare integer variable $rnd2$
    repeat
        $rnd1$ := random integer number between 1 and $k$
        $rnd2$ := random integer number between 1 and $k$
    until $rnd1 \neq rnd2$;
    Declare integer variable $FirstRow$
    Declare integer variable $SecondRow$
    if $rnd1 < rnd2$ then
        $FirstRow := rnd1$
        $SecondRow := rnd2$
    else
        $FirstRow := rnd2$
        $SecondRow := rnd1$
    end
    if Theoretical PLC sequence of FirstRow > Theoretical PLC sequence of
      SecondRow then
        Exchange FirstRow with the SecondRow within clone $c$
    end
end
for $c := 1$ to $LS_{repeat}$ do
    Evaluate clone $c$
end
Order the clones in ascending order by the objective function $T$
if Objective function $T$ of the best clone < Objective function $T$ of the original
  individual then
    Overwrite the original individual by the best clone
end
```

Algorithm 6: Continuous Local Search Procedure

```
Copy the original individual (It will be the initial individual for each clone)
for $c := 1$ to $LS_{repeat}$ do
    Clone the original individual
    Declare integer variable $rnd1$
    Declare integer variable $rnd2$
    repeat
        $rnd1 :=$ random integer number between 1 and $k$
        $rnd2 :=$ random integer number between 1 and $k$
    until $rnd1 \neq rnd2$;
    Declare integer variable $FirstRow$
    Declare integer variable $SecondRow$
    if $rnd1 < rnd2$ then
        $FirstRow := rnd1$
        $SecondRow := rnd2$
    else
        $FirstRow := rnd2$
        $SecondRow := rnd1$
    end
    if Theoretical PLC sequence of FirstRow > Theoretical PLC sequence of SecondRow then
        Exchange FirstRow with the SecondRow within clone $c$
    end
end
for $c := 1$ to $LS_{repeat}$ do
    Evaluate clone $c$
end
Order the clones in ascending order by the objective function $T$
if Objective function $T$ of the best clone < Objective function $T$ of the original individual then
    Overwrite the original individual by the best clone
end
```

Algorithm 7: Independent Local Search Procedure

```
Copy the original individual (It will be the first individual for LS)
for $c := 1$ to $LS_{repeat}$ do
    $c$ will be the copy of that clone, which resulted in the lowest $T$
    Declare integer variable $rnd1$
    Declare integer variable $rnd2$
    repeat
        $rnd1$ := random integer number between 1 and $k$
        $rnd2$ := random integer number between 1 and $k$
    until $rnd1 \neq rnd2$;
    Declare integer variable $FirstRow$
    Declare integer variable $SecondRow$
    if $rnd1 < rnd2$ then
        $FirstRow := rnd1$
        $SecondRow := rnd2$
    else
        $FirstRow := rnd2$
        $SecondRow := rnd1$
    end
    if Theoretical PLC sequence of FirstRow > Theoretical PLC sequence of
      SecondRow then
        Exchange FirstRow with the SecondRow within clone $c$
    end
    Evaluate clone $c$
end
if Objective function $T$ of the best clone < Objective function $T$ of the original
  individual then
    Overwrite the original individual by the best clone
end
```

Algorithm 8: Best Development Local Search Procedure

exchanged. Figure 13 shows a strict LS operator of one clone, where $k = 20$ and the length of the selected PLC homogeneous segment is 4.

3.3.3.5.5 Local Search with Simulated Annealing All previously described 6 LS operator have an alternative, which is completed using the SA method. If Eq. (41) holds, the operator overwrites the original bacterium with the second best clone. Otherwise the best clone is transferred, which can even be equal to the original bacterium. In Eq. (41) $OverwritePortion \sim U\left(0, 100\right)$ is a uniform random number, gen is the current generation, N_{gen} is the total number of generations, and τ_{LS} is a parameter.

$$OverwritePortion < e^{-\frac{gen \cdot \tau_{LS}}{N_{gen}}} \tag{41}$$

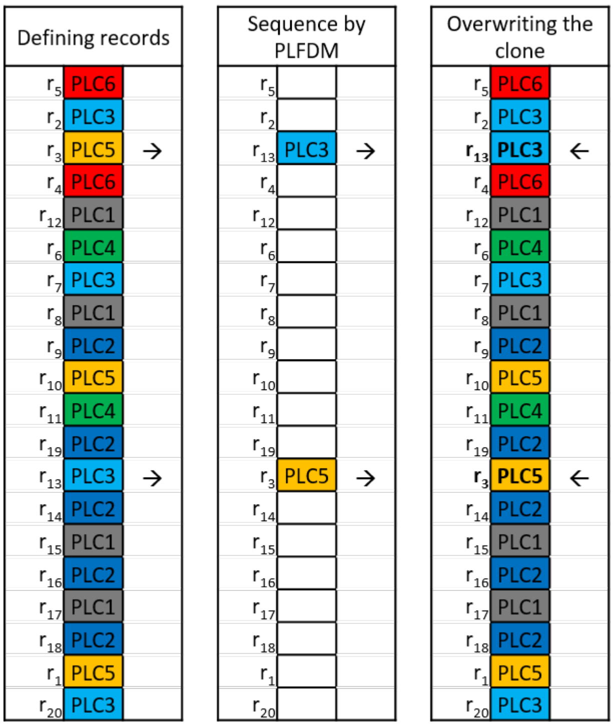
3.3.3.6 Gene Transfer (GT)

After the applied BM and LS operators the gene transfer operator is applied for the whole population. It allows for the recombination of genetic information between two bacteria. First, the population must be divided into two halves. The better 50% of the bacteria are called the superior half, while the other bacteria are the inferior half. One bacterium is randomly selected from the superior half; this will be the source bacterium. Another bacterium is randomly selected from the inferior half; this will be the destination bacterium. A segment from the source bacterium is defined randomly and this segment is used in the destination bacterium. This process is repeated N_{inf} times [32]. In the case of this research different GT operators need to be defined for non-strict and strict alternatives because of their different encoding attributes.

3.3.3.6.1 Non-strict Gene Transfer The non-strict GT (Alg. 9) is applied when a non-strict initial population and BM are used. It randomly defines a coherent L_{GT} long segment in the source bacterium, which is copied into a randomly defined position of the destination bacterium. The previous duplications are deleted in the destination bacterium. Figure 14 illustrates the Non-strict gene transfer of one clone, where $k = 20$, $L_{GT} = 5$, and the perturbed segment is inserted in front of the 18th record.

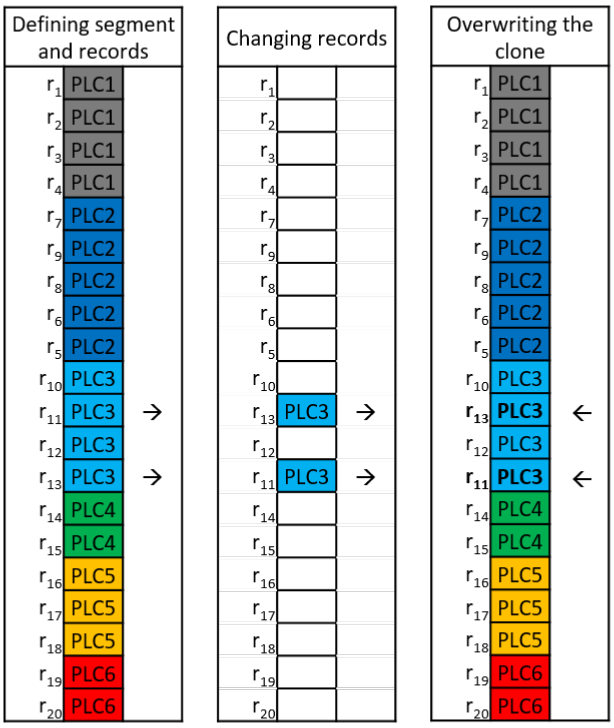
3.3.3.6.2 Strict Gene Transfer The strict GT (Alg. 10) is applied when a strict initial population and BM are used. It randomly defines a PLC homogeneous segment in the source bacterium, which overwrites the same segment of the destination bacterium. Figure 15 depicts the strict gene transfer of one clone, where $k = 20$, L_{GT} of the selected PLC homogeneous segment is 4.

Figure 12: Local search operator in a non-strict case



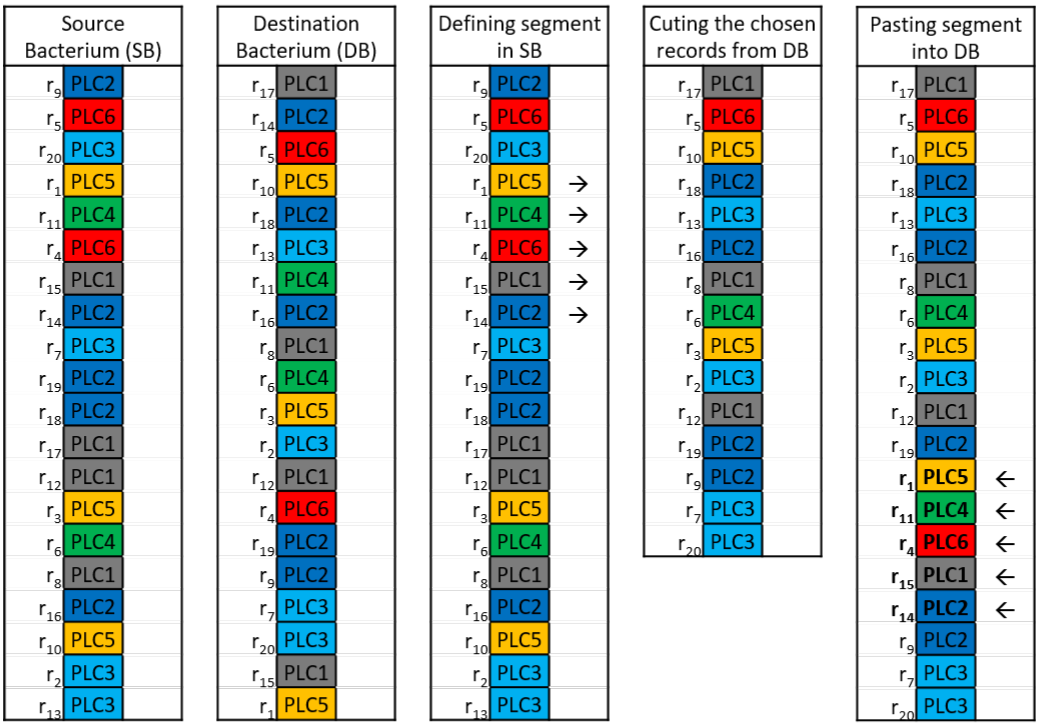
Defining records			Sequence by PLFDM			Overwriting the clone		
r_5	PLC6		r_5			r_5	PLC6	
r_2	PLC3		r_2			r_2	PLC3	
r_3	PLC5	→	r_{13}	PLC3	→	**r_{13}**	**PLC3**	←
r_4	PLC6		r_4			r_4	PLC6	
r_{12}	PLC1		r_{12}			r_{12}	PLC1	
r_6	PLC4		r_6			r_6	PLC4	
r_7	PLC3		r_7			r_7	PLC3	
r_8	PLC1		r_8			r_8	PLC1	
r_9	PLC2		r_9			r_9	PLC2	
r_{10}	PLC5		r_{10}			r_{10}	PLC5	
r_{11}	PLC4		r_{11}			r_{11}	PLC4	
r_{19}	PLC2		r_{19}			r_{19}	PLC2	
r_{13}	PLC3	→	r_3	PLC5	→	**r_3**	**PLC5**	←
r_{14}	PLC2		r_{14}			r_{14}	PLC2	
r_{15}	PLC1		r_{15}			r_{15}	PLC1	
r_{16}	PLC2		r_{16}			r_{16}	PLC2	
r_{17}	PLC1		r_{17}			r_{17}	PLC1	
r_{18}	PLC2		r_{18}			r_{18}	PLC2	
r_1	PLC5		r_1			r_1	PLC5	
r_{20}	PLC3		r_{20}			r_{20}	PLC3	

Figure 13: Local search operator in strict case



Defining segment and records			Changing records			Overwriting the clone		
r_1	PLC1		r_1			r_1	PLC1	
r_2	PLC1		r_2			r_2	PLC1	
r_3	PLC1		r_3			r_3	PLC1	
r_4	PLC1		r_4			r_4	PLC1	
r_7	PLC2		r_7			r_7	PLC2	
r_9	PLC2		r_9			r_9	PLC2	
r_8	PLC2		r_8			r_8	PLC2	
r_6	PLC2		r_6			r_6	PLC2	
r_5	PLC2		r_5			r_5	PLC2	
r_{10}	PLC3		r_{10}			r_{10}	PLC3	
r_{11}	PLC3	→	r_{13}	PLC3	→	**r_{13}**	**PLC3**	←
r_{12}	PLC3		r_{12}			r_{12}	PLC3	
r_{13}	PLC3	→	r_{11}	PLC3	→	**r_{11}**	**PLC3**	←
r_{14}	PLC4		r_{14}			r_{14}	PLC4	
r_{15}	PLC4		r_{15}			r_{15}	PLC4	
r_{16}	PLC5		r_{16}			r_{16}	PLC5	
r_{17}	PLC5		r_{17}			r_{17}	PLC5	
r_{18}	PLC5		r_{18}			r_{18}	PLC5	
r_{19}	PLC6		r_{19}			r_{19}	PLC6	
r_{20}	PLC6		r_{20}			r_{20}	PLC6	

Figure 14: Non-strict Gene Transfer

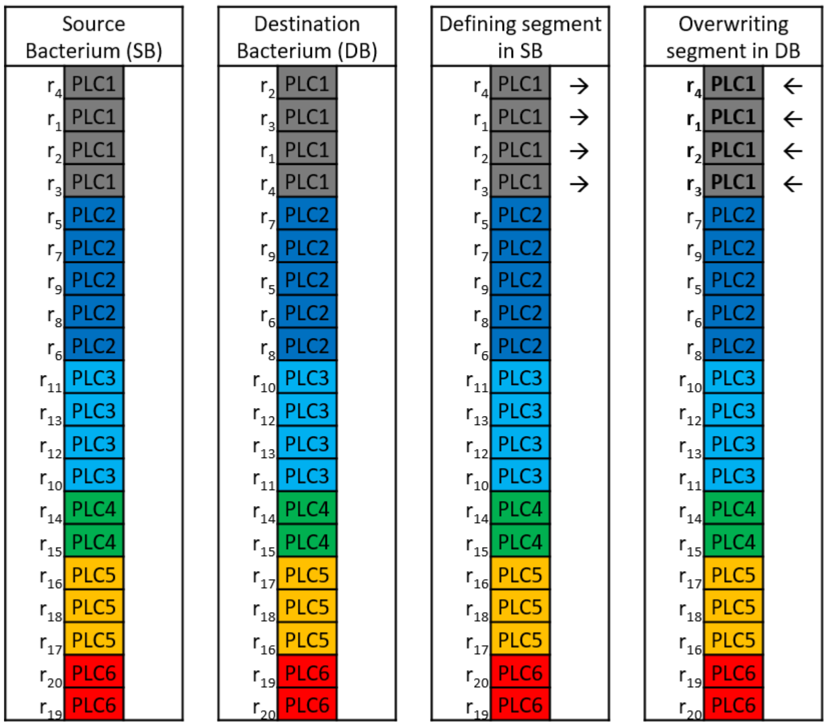


Source Bacterium (SB)	Destination Bacterium (DB)	Defining segment in SB	Cuting the chosen records from DB	Pasting segment into DB
r_9 PLC2	r_{17} PLC1	r_9 PLC2	r_{17} PLC1	r_{17} PLC1
r_5 PLC6	r_{14} PLC2	r_5 PLC6	r_5 PLC6	r_5 PLC6
r_{20} PLC3	r_5 PLC6	r_{20} PLC3	r_{10} PLC5	r_{10} PLC5
r_1 PLC5	r_{10} PLC5	r_1 PLC5 →	r_{18} PLC2	r_{18} PLC2
r_{11} PLC4	r_{18} PLC2	r_{11} PLC4 →	r_{13} PLC3	r_{13} PLC3
r_4 PLC6	r_{13} PLC3	r_4 PLC6 →	r_{16} PLC2	r_{16} PLC2
r_{15} PLC1	r_{11} PLC4	r_{15} PLC1 →	r_8 PLC1	r_8 PLC1
r_{14} PLC2	r_{16} PLC2	r_{14} PLC2 →	r_6 PLC4	r_6 PLC4
r_7 PLC3	r_8 PLC1	r_7 PLC3	r_3 PLC5	r_3 PLC5
r_{19} PLC2	r_6 PLC4	r_{19} PLC2	r_2 PLC3	r_2 PLC3
r_{18} PLC2	r_3 PLC5	r_{18} PLC2	r_{12} PLC1	r_{12} PLC1
r_{17} PLC1	r_2 PLC3	r_{17} PLC1	r_{19} PLC2	r_{19} PLC2
r_{12} PLC1	r_{12} PLC1	r_{12} PLC1	r_9 PLC2	**r_1 PLC5** ←
r_3 PLC5	r_4 PLC6	r_3 PLC5	r_7 PLC3	**r_{11} PLC4** ←
r_6 PLC4	r_{19} PLC2	r_6 PLC4	r_{20} PLC3	**r_4 PLC6** ←
r_8 PLC1	r_9 PLC2	r_8 PLC1		**r_{15} PLC1** ←
r_{16} PLC2	r_7 PLC3	r_{16} PLC2		**r_{14} PLC2** ←
r_{10} PLC5	r_{20} PLC3	r_{10} PLC5		r_9 PLC2
r_2 PLC3	r_{15} PLC1	r_2 PLC3		r_7 PLC3
r_{13} PLC3	r_1 PLC5	r_{13} PLC3		r_{20} PLC3

```
1  //Define a random superior bacterium
2  $Superior = RandomValue(1, N_{ind}/2)$
3  //Define a random inferior bacterium
4  $Inferior = RandomValue(N_{ind}/2 + 1, N_{ind})$
5  Declare integer variable $SegmentStart$ := Randomised definition of the first record
     of the segment $[1, k - L_{GT}]$
6  Define a coherent $L_{GT}$ length segment from $SegmentStart$ record from the
     $Superior$ bacterium
7  Declare integer variable $InsertStart$ := Randomised definition of the record for
     inserting within the $Inferior$ bacterium $[1, k - L_{GT}]$
8  Delete the duplications within the $Inferior$ bacterium
9  Insert segment into the $Inferior$ bacterium in front of the $InsertStart$ record
10 Evaluate the modified individual
11 Overwrite the $Inferior$ bacterium with the modified $Inferior$ bacterium
```

Algorithm 9: Non-strict Gene Transfer Procedure

Figure 15: Strict Gene Transfer



```
//Define a random superior bacterium
$Superior = RandomValue(1, N_{ind}/2)$
//Define a random inferior bacterium
$Inferior = RandomValue(N_{ind}/2 + 1, N_{ind})$
Define a segment from the Superior bacterium with random selection of a PLC
  coherent segment
Overwrite the selected segment in the Inferior bacterium with the selected
  segment from the Superior bacterium
Evaluate the modified individual
Overwrite the Inferior bacterium with the modified Inferior bacterium
```

Algorithm 10: Strict Gene Transfer Procedure

3.3.4 Simulated Annealing algorithm for OPRP-PLF

As part of my research Simulated Annealing (SA) methodology based optimisation algorithms have been developed for the OPRP-PLF for comparison reasons. This well-known optimisation algorithm has often been used to evaluate developed algorithm results on a standard basis [24]. I generalised the method for population based Simulated Annealing algorithms [11].

Since the widely used nature inspired optimisation algorithms such as PSO, DE, ABC and so on are useful for continuous optimisation problem, they are not the best choices for the proposed combinatorial OPRP-PLF. Simulated annealing is applied, because it is effectively applicable for combinatorial problems. The reasons why the SA algorithm is applied, are its potential application in an evolutionary based search as proposed in this research in conjunction with BMA, its possibility to avoid local optimum, and the good experiences about its effectiveness and simplicity ([11]).

The PLF based SA Algorithm encodes the initial population on the same basis as BMA. It has non-strict and strict operator for defining N_{ind} number of initial order picking sequencing alternatives. In the non-strict case the algorithm defines one strict, PLC sequenced eugenic individual. The non-strict and strict SA operators run N_{gen} times for each individual.

3.3.4.1 Non-strict Simulated Annealing Operator

The non-strict SA operator (Alg. 11) randomly selects S numbers of different picking positions from the individual to define an S long segment. $S \in [2, IterationPortion]$, where $IterationPortion \in \mathbb{N}$ is based on Eq. (42). The defined segment is randomly perturbed and transferred into the individual's clone.

$$IterationPortion = \left\lfloor 1 - \frac{gen}{N_{gen}} \right\rceil \cdot k \tag{42}$$

The overwriting term is based on SA methodology except for the eugenic individual. The SA term is neglected in the case of the eugenic individual to evaluate SA efficiency. The original eugenic individual is overwritten when its clone has a lower lead time.

If the perturbed non-eugenic clone has a higher lead time than the original individual but Eq. (41) holds, the original individual is overwritten by the perturbed clone.

3.3.4.2 Strict Simulated Annealing Operator

Each strict, PLC based segment of the individual is perturbed separately by the strict SA operator (Alg. 12). It randomly defines the S long segment of each PLC for perturbation. $S \in [2, IterationPortion]$, where $IterationPortion \in \mathbb{N}$ is based on Eq. (43), where $k_{PLCsegment}$ is the length of the PLC based segment, which contains every picking position.

```
Execute Non-strict initial population generation method
for $i := 1$ to $N_{ind}$ do
    Evaluation of each individual by objective function $T$
end
for $g := 1$ to $N_{gen}$ do
    for $i := 1$ to $N_{ind}$ do
        Declare $S$ integer variable := Randomised definition of the segment length,
          $S \in [2, IterationPortion]$, where $IterationPortion \in \mathbb{N}$ is based on
          Eq. (42).
        for $j := 1$ to $S$ do
            Randomly allocate an unselected record to the segment
        end
        Perturb the incoherent segment
        Overwrite the selected records by the perturbed values
        Evaluate the individual
        if The individual is eugenic = true then
            if Objective function T of the modified individual < Objective function T
              of the original individual then
                Overwrite the original individual by the modified individual
            end
        else
            if (Objective function T of the modified individual < Objective function
              T of the original individual) or Equation (40) holds then
                Overwrite the original individual by the modified individual
            end
        end
        Order the population in ascending order by the objective function $T$
    end
end
```

Algorithm 11: Non-strict Simulated Annealing Procedure

Each perturbation segment is randomly perturbed and transferred into the individual clone. After perturbation the individuals are evaluated and the SA based overwriting term is applied, which is described in the previous section.

$$IterationPortion = \left\lfloor 1 - \frac{gen}{N_{gen}} \right\rceil \cdot k_{PLCsegment} \tag{43}$$

```
1  Execute strict initial population generation method
2  for $i := 1$ to $N_{ind}$ do
3      Evaluation of each individual by objective function $T$
4  end
5  for $g := 1$ to $N_{gen}$ do
6      for $i := 1$ to $N_{ind}$ do
7          Declare $S$ integer variable := Randomised definition the segment length,
             $S \in [2, IterationPortion]$, where $IterationPortion \in \mathbb{N}$ is based on
             Eq. (43).
8          for $p := 1$ to Number of PLCs within the list do
9              for $j := 1$ to $S$ do
10                 Randomly allocate an unselected record from the records of PLC $p$
                     to the segment
11             end
12             Perturb the PLC homogeneous segment
13             Overwrite the selected records with the perturbed values
14         end
15         Evaluate the individual
16         if (Objective function $T$ of the modified individual $<$ Objective function $T$ of
             the original individual) or Equation (40) holds then
17             Overwrite the original individual by the modified individual
18         end
19         Order the population in ascending order by the objective function $T$
20     end
21 end
```

Algorithm 12: Strict Simulated Annealing Procedure

3.3.5 Summarising statements

While the algorithm running time is a critical factor of the at least exponential OPRP-PLF, I offer the Bacterial Memetic Algorithm (BMA) for the defined problem, because it has already been successfully utilised in several combinatorial optimisation problems because of its fast convergence speed.

Thesis statement 3. *While the order picking routing optimisation objective functions usually consider the travelling and the picking time, I specified an objective function for minimising the lead time of the OPRP-PLF, which besides those aspects considers the reconstruction time too.*
Based on analytic examination I justified the necessity of the algorithm development, showing that since the shortest route could cause more reconstruction and higher lead time because of the Pallet Loading Feature, hence the picker should be supported by algorithms in the case of complex OPRP-PLF.
I constructed Bacterial Memetic Algorithm (BMA) and population based Simulated Annealing (SA) algorithms for comparing two cases, when reconstruction is avoided (Strict process) and when it is allowed (Non-strict process). I combined the bacterial mutation and the local search operators of the BMA with SA algorithms as a BMA novelty to increase the optimisation efficiency within the short optimisation time.

My publications related to the statement: [11], [9].

3.4 Algorithm evaluation based on different level of complexity order picking lists

I defined possible algorithms for OPRP-PLF with the possible combinations of the above-mentioned operators. The algorithms are implemented and the test environment is modelled in Plant Simulation environment. The algorithms are evaluated and compared on the same basis with the computer simulation model.

Plant Simulation supports to create digital models of logistics systems and it validates the right implementation through visualised movements. The object oriented methodology, the programmability, and the predefined functions made the environment a valuable tool for logistics system modelling, evaluating, and development. Since Plant Simulation is a well known tool for me, it has been applied for algorithm development and evaluation. However the defined architecture and algorithms are reproducible in any other simulation environment, framework or programming language. The algorithms have been generalised for possible reproduction, because besides Plant Simulation is a world wide known environment it might not be widespread and it requires licences. Furthermore, the possible industrial application of the solution also makes the system-independent, generalised description important [4].

3.4.1 Simulation environment and functionality

I created a Plant Simulation based test environment to let me able to evaluate several algorithm alternatives based on various inputs (Layout, SLA, orders). The alternatives are defined based on combinations of several optimisation operator and parameters values.

I developed wide functionality for the input specification to make me able to evaluate the behaviour of the algorithms based on different inputs. It is possible to define the geometry of the warehouse layout flexibly, where the picking positions and the routes are movable and variable long objects. The length of the route objects are scalable for real layouts and those are changeable with a multiplier variable to proportionately define the layout dimensions. It makes possible to specify a standardise layout, increase the layout dimensions and route length proportionately to evaluate the algorithms in the case of smaller and bigger layouts (shorter and longer picking length). It was necessary to prepare the test environment for defining different SLA-s (Storage Location Assignment) to evaluate those effects on the picking effectiveness. At the actual state of my research, these SLA-s should be previously defined manually based on industrially relevant logic or methodology as it was discussed in Section 2.1.2. The uploaded SLA input contains the products and the order picking position ID. Furthermore, I would like to evaluate the algorithms based on different complexity order picking lists. Defining order picking lists with arbitrary item combinations and length make me able to examine algorithms based on industrially relevant order picking lists with specific nature. The uploaded order picking lists contain the ordered items and the ordered quantity. The PC and POPC parameter of the order picking records are defined based on the product database and the classification methods.

Besides the input specification, the algorithm parameters' specification is a considerable function of the test environments. It makes me able to define each parameter of the algorithms and the algorithm operators in one easily manageable and embraceable table. It defines each alternative for evaluation based on the algorithm operator combination, the operator parameter values, the global optimisation parameters (e.g.: random seeds, number of generations, number of individuals), and the applied layout and SLA inputs. The defined alternatives will be discussed in Section 3.4.2.

Furthermore, I developed wide functionality for algorithm evaluation. The model collects data during the optimisation procedure about each alternative for further analysis. The main basic data for alternatives are the results of the last bacteria of the last generation (time results based on the objective function and picking sequence) and the best results for each generation. I developed analysis functionality for the test environment to make me able to compare alternatives based on calculated indicators and graphs based on the collected data. The results, the indicators, and the graphs will be discussed in Section 3.4.3.

For validation purpose I developed some visualisation tools to be able to overlook and verify the resulted order picking sequence on the layout. The test environment marks the order picking positions based on PC colour code and visualises the picking sequence with arrows. The relevant applications will be discussed in Section 3.5.4.

3.4.2 Algorithms and Parameters

The operator combinations define the possible algorithms, which are summarised in Table 16. The non-strict initial population is combined only with non-strict operators, without any strict method. The strict initial population is combined with both non-strict and strict BM operators to evaluate the optimisation of pre-sequenced but non-strict solutions. Non-strict BM operators used only non-strict LS and GT operators, furthermore strict BM operators used only strict LS and GT operators.

Table 17 describes the parameter values of the mentioned algorithms, which were defined based on previous tests. The operators apply the same parameter values for each algorithm to ensure the same basis for the algorithm evaluation and behaviour comparison. The aim was to define the number of generations (N_{gen}) to reach an approximately equal budget for each algorithm to ensure the fair comparison of the algorithms. The budget is the number of objective function evaluations. Table 17 shows the N_{gen} values, which were necessary to achieve acceptable results for a 20 record long order picking list. Their

Table 16: Algorithm alternatives - combinations of operators

Algorithm	Initial Population	Bacterial Mutation	Local Search	Gene Transfer	SA
NonStrictInit+NonStrictBM+ContLS	Non-Strict	Non-Strict	Continuous	Non-Strict	
NonStrictInit+NonStrictBM+ContLS-SA	Non-Strict	Non-Strict	Continuous with SA	Non-Strict	
NonStrictInit+NonStrictBM+IndepLS	Non-Strict	Non-Strict	Independent	Non-Strict	
NonStrictInit+NonStrictBM+IndepLS-SA	Non-Strict	Non-Strict	Independent with SA	Non-Strict	
NonStrictInit+NonStrictBM+BestDevLS	Non-Strict	Non-Strict	Best Development	Non-Strict	
NonStrictInit+NonStrictBM+BestDevLS-SA	Non-Strict	Non-Strict	Best Development with SA	Non-Strict	
NonStrictInit+NonStrictBM-SA+ContLS	Non-Strict	Non-Strict with SA	Continuous	Non-Strict	
NonStrictInit+NonStrictBM-SA+ContLS-SA	Non-Strict	Non-Strict with SA	Continuous with SA	Non-Strict	
NonStrictInit+NonStrictBM-SA+IndepLS	Non-Strict	Non-Strict with SA	Independent	Non-Strict	
NonStrictInit+NonStrictBM-SA+IndepLS-SA	Non-Strict	Non-Strict with SA	Independent with SA	Non-Strict	
NonStrictInit+NonStrictBM-SA+BestDevLS	Non-Strict	Non-Strict with SA	Best Development	Non-Strict	
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	Non-Strict	Non-Strict with SA	Best Development with SA	Non-Strict	
StrictInit+NonStrictBM+ContLS	Strict	Non-Strict	Continuous	Non-Strict	
StrictInit+NonStrictBM+ContLS-SA	Strict	Non-Strict	Continuous with SA	Non-Strict	
StrictInit+NonStrictBM+IndepLS	Strict	Non-Strict	Independent	Non-Strict	
StrictInit+NonStrictBM+IndepLS-SA	Strict	Non-Strict	Independent with SA	Non-Strict	
StrictInit+NonStrictBM+BestDevLS	Strict	Non-Strict	Best Development	Non-Strict	
StrictInit+NonStrictBM+BestDevLS-SA	Strict	Non-Strict	Best Development with SA	Non-Strict	
StrictInit+NonStrictBM-SA+ContLS	Strict	Non-Strict with SA	Continuous	Non-Strict	
StrictInit+NonStrictBM-SA+ContLS-SA	Strict	Non-Strict with SA	Continuous with SA	Non-Strict	
StrictInit+NonStrictBM-SA+IndepLS	Strict	Non-Strict with SA	Independent	Non-Strict	
StrictInit+NonStrictBM-SA+IndepLS-SA	Strict	Non-Strict with SA	Independent with SA	Non-Strict	
StrictInit+NonStrictBM-SA+BestDevLS	Strict	Non-Strict with SA	Best Development	Non-Strict	
StrictInit+NonStrictBM-SA+BestDevLS-SA	Strict	Non-Strict with SA	Best Development with SA	Non-Strict	
StrictInit+StrictBM+StrictContLS	Strict	Strict	Strict Continuous	Strict	
StrictInit+StrictBM+StrictContLS-SA	Strict	Strict	Strict Continuous with SA	Strict	
StrictInit+StrictBM+StrictIndepLS	Strict	Strict	Strict Independent	Strict	
StrictInit+StrictBM+StrictIndepLS-SA	Strict	Strict	Strict Independent with SA	Strict	
StrictInit+StrictBM+StrictBestDevLS	Strict	Strict	Strict Best Development	Strict	
StrictInit+StrictBM+StrictBestDevLS-SA	Strict	Strict	Strict Best Development with SA	Strict	
StrictInit+StrictBM-SA+StrictContLS	Strict	Strict with SA	Strict Continuous	Strict	
StrictInit+StrictBM-SA+StrictContLS-SA	Strict	Strict with SA	Strict Continuous with SA	Strict	
StrictInit+StrictBM-SA+StrictIndepLS	Strict	Strict with SA	Strict Independent	Strict	
StrictInit+StrictBM-SA+StrictIndepLS-SA	Strict	Strict with SA	Strict Independent with SA	Strict	
StrictInit+StrictBM-SA+StrictBestDevLS	Strict	Strict with SA	Strict Best Development	Strict	
StrictInit+StrictBM-SA+StrictBestDevLS-SA	Strict	Strict with SA	Strict Best Development with SA	Strict	
SA	Non-Strict for SA				SA
StrictSA	Strict for SA				Strict SA

Table 17: Parameters of algorithm alternatives

Algorithm	$Eugenic$	N_{gen}	N_{ind}	N_{clones}	$L_{segment}$	LS_{repeat}	N_{inf}	L_{GT}	τ	τ_{LS}
NonStrictInit+NonStrictBM+ContLS	true	20	50	10	4	5	20	5		
NonStrictInit+NonStrictBM+ContLS-SA	true	20	50	10	4	5	20	5		15
NonStrictInit+NonStrictBM+IndepLS	true	20	50	10	4	5	20	5		
NonStrictInit+NonStrictBM+IndepLS-SA	true	20	50	10	4	5	20	5		15
NonStrictInit+NonStrictBM+BestDevLS	true	20	50	10	4	5	20	5		
NonStrictInit+NonStrictBM+BestDevLS-SA	true	20	50	10	4	5	20	5		15
NonStrictInit+NonStrictBM-SA+ContLS	true	20	50	10	4	5	20	5	15	
NonStrictInit+NonStrictBM-SA+ContLS-SA	true	20	50	10	4	5	20	5	15	15
NonStrictInit+NonStrictBM-SA+IndepLS	true	20	50	10	4	5	20	5	15	
NonStrictInit+NonStrictBM-SA+IndepLS-SA	true	20	50	10	4	5	20	5	15	15
NonStrictInit+NonStrictBM-SA+BestDevLS	true	20	50	10	4	5	20	5	15	
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	true	20	50	10	4	5	20	5	15	15
StrictInit+NonStrictBM+ContLS	false	20	50	10	4	5	20	5		
StrictInit+NonStrictBM+ContLS-SA	false	20	50	10	4	5	20	5		15
StrictInit+NonStrictBM+IndepLS	false	20	50	10	4	5	20	5		
StrictInit+NonStrictBM+IndepLS-SA	false	20	50	10	4	5	20	5		15
StrictInit+NonStrictBM+BestDevLS	false	20	50	10	4	5	20	5		
StrictInit+NonStrictBM+BestDevLS-SA	false	20	50	10	4	5	20	5		15
StrictInit+NonStrictBM-SA+ContLS	false	20	50	10	4	5	20	5	15	
StrictInit+NonStrictBM-SA+ContLS-SA	false	20	50	10	4	5	20	5	15	15
StrictInit+NonStrictBM-SA+IndepLS	false	20	50	10	4	5	20	5	15	
StrictInit+NonStrictBM-SA+IndepLS-SA	false	20	50	10	4	5	20	5	15	15
StrictInit+NonStrictBM-SA+BestDevLS	false	20	50	10	4	5	20	5	15	
StrictInit+NonStrictBM-SA+BestDevLS-SA	false	20	50	10	4	5	20	5	15	15
StrictInit+StrictBM+StrictContLS	false	17	50	10		5	20			
StrictInit+StrictBM+StrictContLS-SA	false	17	50	10		5	20			15
StrictInit+StrictBM+StrictIndepLS	false	17	50	10		5	20			
StrictInit+StrictBM+StrictIndepLS-SA	false	17	50	10		5	20			15
StrictInit+StrictBM+StrictBestDevLS	false	17	50	10		5	20			
StrictInit+StrictBM+StrictBestDevLS-SA	false	17	50	10		5	20			15
StrictInit+StrictBM-SA+StrictContLS	false	17	50	10		5	20		15	
StrictInit+StrictBM-SA+StrictContLS-SA	false	17	50	10		5	20		15	15
StrictInit+StrictBM-SA+StrictIndepLS	false	17	50	10		5	20		15	
StrictInit+StrictBM-SA+StrictIndepLS-SA	false	17	50	10		5	20		15	15
StrictInit+StrictBM-SA+StrictBestDevLS	false	17	50	10		5	20		15	
StrictInit+StrictBM-SA+StrictBestDevLS-SA	false	17	50	10		5	20		15	15
SA	true	1200	50						10	
StrictSA	false	1200	50						10	

budgets are shown in Table 18 for each algorithm and their operators. The value of N_{gen} parameter is defined for the order picking lists based on their complexity. Longer and more complex lists require more generations (N_{gen}) because of the exponential growth of the possible combinations. It is recommended to use less generation for simple lists to decrease unnecessary evaluations and to keep the comparison fair.

Figure 16 depicts the order picking zone layout of the test environment with the picker. The colleague walks from position to position (rectangles) with the unit load via the footpath (grey line). The grey rectangle visualises the departure and arrival position. 480 different products are allocated on 480 order picking positions with totally randomised SLA. The layout is scaled, but the real distances between the position objects are doubled to model a realistically huge warehouse (1920 picking positions), where a complex stacking attribute based routing algorithm would be necessary. The doubled distances between position midpoints within the aisle is 1.8 metres and between aisles is 10.6 metres. The model uses a simple symmetrical and triangular PLFDM (Table 19).

Three order picking lists have been defined with a different number of records (10, 20, and 50 records). Its PLC occurrence has been summarised in Table 20. The ordered quantity of each record is ignored for simplification. The model assumes that the ordered quantity of each record equals exactly 1 layer of products on the UL.

To support the possible reproduction the necessary inputs are attached as appendices. The Appendix section contains the layout definition, the details of the applied items with SLA (order picking position), and the contents of the applied O10, O20, and O50 orders.

As it was discussed before, the N_{gen} value should be matched to the complexity of the order picking list in order to have a fair algorithm comparison. The possible combinations of the order picking sequence could highlight the complexity of O10, O20, and O50 picking lists. When the algorithm applied only strict operators, the number of possible combinations is defined by Eq. (21). When reconstruction is allowed (one of the non-strict operators is applied), $k!$ defines the number of possible combinations. This number can help define the necessary number of generations, as it is shown in Table 21. The different number of handled segments causes a different N_{gen} value for non-strict and strict BMA algorithms. Furthermore, BMA and SA algorithms need different N_{gen} values due to their different methodology. The budget column in Table 21 shows the approximated budget of the algorithms.

Table 22 shows the parameter values of the previously described objective function. The distances are defined by the scaled layout.

3.4.3 Results and Evaluation

The O10 and O20 lists have been optimised by every algorithm to define the most effective operator combinations. The algorithms are evaluated and compared on the same basis

Table 18: Number of evaluations during the optimisation of a 20 record long order picking list

Algorithm	Sum	Init	BM	LS	GT	SA
NonStrictInit+NonStrictBM+ContLS	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM+ContLS-SA	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM+IndepLS	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM+IndepLS-SA	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM+BestDevLS	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM+BestDevLS-SA	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM-SA+ContLS	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM-SA+ContLS-SA	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM-SA+IndepLS	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM-SA+IndepLS-SA	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM-SA+BestDevLS	60450	50	55000	5000	400	0
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	60450	50	55000	5000	400	0
StrictInit+NonStrictBM+ContLS	60450	50	55000	5000	400	0
StrictInit+NonStrictBM+ContLS-SA	60450	50	55000	5000	400	0
StrictInit+NonStrictBM+IndepLS	60450	50	55000	5000	400	0
StrictInit+NonStrictBM+IndepLS-SA	60450	50	55000	5000	400	0
StrictInit+NonStrictBM+BestDevLS	60450	50	55000	5000	400	0
StrictInit+NonStrictBM+BestDevLS-SA	60450	50	55000	5000	400	0
StrictInit+NonStrictBM-SA+ContLS	60450	50	55000	5000	400	0
StrictInit+NonStrictBM-SA+ContLS-SA	60450	50	55000	5000	400	0
StrictInit+NonStrictBM-SA+IndepLS	60450	50	55000	5000	400	0
StrictInit+NonStrictBM-SA+IndepLS-SA	60450	50	55000	5000	400	0
StrictInit+NonStrictBM-SA+BestDevLS	60450	50	55000	5000	400	0
StrictInit+NonStrictBM-SA+BestDevLS-SA	60450	50	55000	5000	400	0
StrictInit+StrictBM+StrictContLS	60740	50	56100	4250	340	0
StrictInit+StrictBM+StrictContLS-SA	60740	50	56100	4250	340	0
StrictInit+StrictBM+StrictIndepLS	60740	50	56100	4250	340	0
StrictInit+StrictBM+StrictIndepLS-SA	60740	50	56100	4250	340	0
StrictInit+StrictBM+StrictBestDevLS	60740	50	56100	4250	340	0
StrictInit+StrictBM+StrictBestDevLS-SA	60740	50	56100	4250	340	0
StrictInit+StrictBM-SA+StrictContLS	60740	50	56100	4250	340	0
StrictInit+StrictBM-SA+StrictContLS-SA	60740	50	56100	4250	340	0
StrictInit+StrictBM-SA+StrictIndepLS	60740	50	56100	4250	340	0
StrictInit+StrictBM-SA+StrictIndepLS-SA	60740	50	56100	4250	340	0
StrictInit+StrictBM-SA+StrictBestDevLS	60740	50	56100	4250	340	0
StrictInit+StrictBM-SA+StrictBestDevLS-SA	60740	50	56100	4250	340	0
SA	60050	50	0	0	0	60000
StrictSA	60050	50	0	0	0	60000

Table 19: Pallet Loading Feature based Decision Matrix (PLFDM) of the test environment

	Bag	*Pail*	*Can*	*BigBox*	*SmallBox*	*Fragile*
Bag	1	1	1	1	1	1
Pail	0	1	1	1	1	1
Can	0	0	1	1	1	1
BigBox	0	0	0	1	1	1
SmallBox	0	0	0	0	1	1
Fragile	0	0	0	0	0	1

Figure 16: Order picking zone layout in Plant Simulation environment

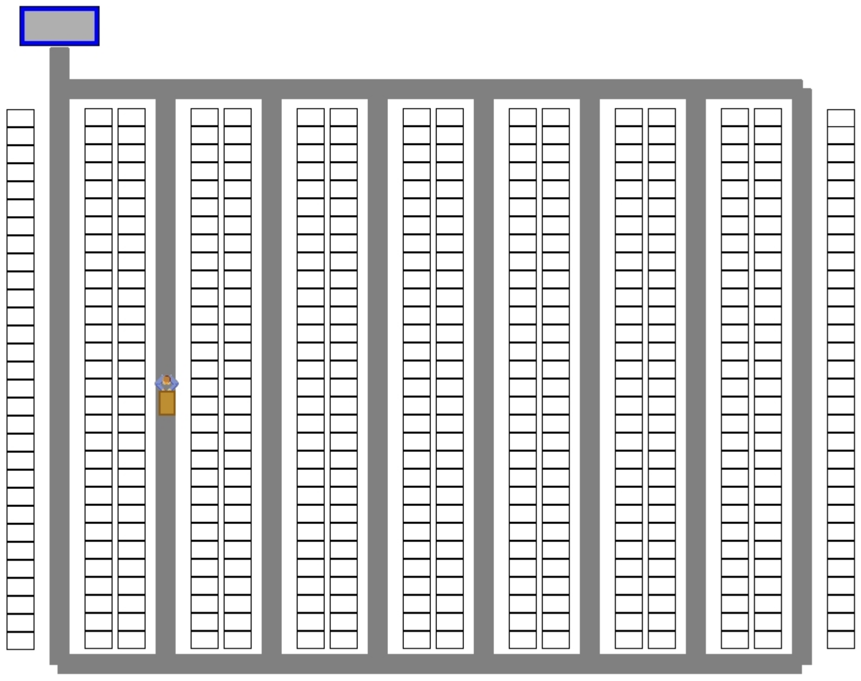


Table 20: PLC occurrence of the optimised order picking lists

	O10	O20	O50
Bag	0	3	8
Pail	1	1	9
Can	2	1	7
BigBox	3	7	12
SmallBox	4	7	12
Fragile	0	1	2

Table 21: N_{gen} values of each order picking list

Encoding method	Order	Possible combinations	Non-strict N_{gen}	Strict N_{gen}	SA N_{gen}	Budget
Strict	O10	$2.88 \cdot 10^2$	5	4	192	9600
Strict	O20	$1.52 \cdot 10^8$	20	17	1200	60 000
Strict	O50	$3.38 \cdot 10^{31}$	140	280	20 000	1 000 000
Non-Strict	O10	$3.63 \cdot 10^6$	5	4	192	9600
Non-Strict	O20	$2.43 \cdot 10^{18}$	20	17	1200	60 000
Non-Strict	O50	$3.04 \cdot 10^{64}$	140	280	20 000	1 000 000

Table 22: Parameter values of the objective function

Parameter	Value
α	1
β	1
γ	1
v	1.67 m/s
t_P	10 sec
t_R	7.5 sec

by the simulation model. Each algorithm is used 10 times with 10 different seeds due to their stochastic nature. The 10 seed results of each algorithm for O10 are summarised in Table 23. Table 24 shows the results of O20. BestEugLT and BestNotEugLT columns display the lowest eugenic and not eugenic results of the seeds where those are relevant. BestLT shows the lowest lead time of the 10 algorithm tests. The AVG column is the average lead time and the SD is the standard deviation of the 10 seed results. The BestOcc column shows how many times the algorithm reached the BestLT result during the 10 seeds. The 1stBestGen% column defines the first generation, which reached the BestLT and it shows the percentage of the total number of generations.

Table 23 and Table 24 are ordered by the BestLT. These tables show that the strict PLC based cases (when each operator is strict) resulted in higher lead times than the non-strict algorithms. The non-strict algorithms reached the lowest lead time (7:04.6707) for O1. It performed the following objective function results:

- T_T (travel time) = 4:24.6707
- T_P (picking time) = 1:40.0000
- T_R (reconstruction) = 1:00.0000

The best strict objective function result is 7:24.3114 with the following objective function parameters:

- T_T (travel time) = 5:44.3114
- T_P (picking time) = 1:40.0000
- T_R (reconstruction) = 0:00.0000

The objective function results highlight the necessity of reconstruction, which was needed to reach the best lead time. The reconstruction made it possible for the algorithm to find a picking sequence with shorter travel time. While the reconstruction time is less than the travel time savings, the non-strict sequence will cause a lower lead time than the strict one.

However, if a seed of an algorithm achieved a low lead time, the algorithm might not perform well in every seed case due to high SD. In Table 25 and Table 26 the results are listed in increasing order of SD. The results show that the strict cases performed well from this point of view, which highlights the good performance of these algorithms. The algorithms of the low lead times are in the mid-range of the SD list.

In parallel to the previous mentioned indicators, the most meaningful evaluation indicator could be the average of the seeds. Table 27 and Table 28 are ordered by the AVG column, which highlights the most reliable algorithms. The non-strict algorithms, which allow reconstruction during order picking, resulted in a definitely lower lead time than

Table 23: Results of O10 optimisation - sequenced by best lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM+ContLS-SA	–	7:04.6707	7:04.6707	7:05.9162	2.6258	8	20
StrictInit+NonStrictBM+BestDevLS-SA	–	7:04.6707	7:04.6707	7:06.0778	4.4499	9	40
StrictInit+NonStrictBM-SA+BestDevLS-SA	–	7:04.6707	7:04.6707	7:06.3323	3.6379	8	60
StrictInit+NonStrictBM-SA+IndepLS	–	7:04.6707	7:04.6707	7:06.3323	3.6379	8	60
SA	7:20.3593	7:04.6707	7:04.6707	7:07.5329	4.8182	7	59
StrictInit+NonStrictBM+IndepLS-SA	–	7:04.6707	7:04.6707	7:08.0419	7.2274	8	20
StrictInit+NonStrictBM-SA+IndepLS-SA	–	7:04.6707	7:04.6707	7:08.7335	5.4766	6	60
StrictInit+NonStrictBM+BestDevLS	–	7:04.6707	7:04.6707	7:08.9192	7.6023	7	20
StrictInit+NonStrictBM-SA+ContLS	–	7:04.6707	7:04.6707	7:09.0329	6.6549	6	20
StrictInit+NonStrictBM-SA+ContLS-SA	–	7:04.6707	7:04.6707	7:09.1946	5.9364	6	20
NonStrictInit+NonStrictBM-SA+ContLS	7:32.8443	7:04.6707	7:04.6707	7:09.5629	7.9720	7	40
NonStrictInit+NonStrictBM+ContLS	7:04.6707	7:04.6707	7:04.6707	7:09.7036	7.2816	6	40
NonStrictInit+NonStrictBM+IndepLS-SA	7:10.8982	7:04.6707	7:04.6707	7:09.8174	4.8474	4	60
NonStrictInit+NonStrictBM-SA+BestDevLS	7:04.6707	7:04.6707	7:04.6707	7:10.1407	6.0901	5	40
NonStrictInit+NonStrictBM-SA+IndepLS	7:26.6168	7:04.6707	7:04.6707	7:10.1407	8.1321	6	60
NonStrictInit+NonStrictBM-SA+IndepLS-SA	7:24.3114	7:04.6707	7:04.6707	7:10.2335	5.1214	4	60
StrictInit+NonStrictBM+ContLS	–	7:04.6707	7:04.6707	7:10.4222	8.3243	6	40
NonStrictInit+NonStrictBM+ContLS-SA	7:10.8982	7:04.6707	7:04.6707	7:11.4790	5.2045	3	60
StrictInit+NonStrictBM-SA+BestDevLS	–	7:04.6707	7:04.6707	7:11.9910	6.8781	4	80
NonStrictInit+NonStrictBM-SA+ContLS-SA	7:26.6168	7:04.6707	7:04.6707	7:12.3084	7.6995	4	60
NonStrictInit+NonStrictBM+IndepLS	7:24.3114	7:04.6707	7:04.6707	7:12.4371	9.3769	5	40
StrictInit+NonStrictBM+IndepLS	–	7:04.6707	7:04.6707	7:13.0749	8.6491	4	40
NonStrictInit+NonStrictBM+BestDevLS-SA	7:42.6048	7:04.6707	7:04.6707	7:13.0958	6.6525	3	60
NonStrictInit+NonStrictBM+BestDevLS	7:32.8443	7:04.6707	7:04.6707	7:13.2335	9.2154	5	60
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	7:22.9042	7:04.6707	7:04.6707	7:14.9910	10.7882	4	60
StrictInit+StrictBM+StrictBestDevLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictBestDevLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictContLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictContLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictIndepLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictIndepLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictBestDevLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictBestDevLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictContLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictContLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictIndepLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictIndepLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictSA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	1

Table 24: Results of O20 optimisation - sequenced by best lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM+IndepLS	–	11:51.1976	11:51.1976	12:11.9731	10.8084	1	100
NonStrictInit+NonStrictBM+ContLS-SA	11:51.1976	11:56.1377	11:51.1976	12:20.4970	20.4119	1	65
StrictInit+NonStrictBM+ContLS	–	11:51.3473	11:51.3473	12:09.7784	13.9960	1	85
StrictInit+NonStrictBM+IndepLS-SA	–	11:51.3473	11:51.3473	12:10.4671	14.4946	1	80
NonStrictInit+NonStrictBM+IndepLS	11:51.3473	12:09.0419	11:51.3473	12:17.0599	12.0535	1	90
StrictInit+NonStrictBM+BestDevLS	–	11:53.4431	11:53.4431	12:11.0210	10.7707	1	80
NonStrictInit+NonStrictBM+IndepLS-SA	12:12.1257	11:53.4431	11:53.4431	12:18.5329	15.2699	1	80
StrictInit+NonStrictBM-SA+ContLS	–	11:53.5030	11:53.5030	12:07.7695	9.4708	1	95
NonStrictInit+NonStrictBM-SA+BestDevLS	12:04.8204	11:53.5030	11:53.5030	12:17.0269	14.7494	1	90
StrictInit+NonStrictBM-SA+ContLS-SA	–	11:55.2695	11:55.2695	12:12.8263	11.8230	1	55
StrictInit+NonStrictBM+ContLS-SA	–	11:56.1377	11:56.1377	12:13.3204	9.0991	1	85
StrictInit+NonStrictBM+BestDevLS-SA	–	11:56.5868	11:56.5868	12:08.6228	8.3947	1	100
NonStrictInit+NonStrictBM-SA+ContLS	12:22.4251	11:56.5868	11:56.5868	12:21.1886	14.8191	1	95
StrictInit+NonStrictBM-SA+BestDevLS-SA	–	11:56.9760	11:56.9760	12:17.2904	10.7730	1	80
StrictInit+NonStrictBM-SA+IndepLS-SA	–	11:58.3533	11:58.3533	12:17.7335	14.4560	1	90
NonStrictInit+NonStrictBM+BestDevLS-SA	12:02.3054	11:58.4431	11:58.4431	12:14.2006	11.9146	1	100
NonStrictInit+NonStrictBM+ContLS	12:17.0659	11:58.4431	11:58.4431	12:19.1317	12.8389	1	80
StrictInit+NonStrictBM-SA+BestDevLS	–	11:58.5928	11:58.5928	12:18.4042	10.2987	1	100
NonStrictInit+NonStrictBM-SA+ContLS-SA	11:58.5928	12:19.8204	11:58.5928	12:27.5928	14.1030	1	50
StrictInit+NonStrictBM-SA+IndepLS	–	12:04.9102	12:04.9102	12:17.6497	9.6629	1	95
NonStrictInit+NonStrictBM-SA+IndepLS-SA	12:31.4671	12:06.9162	12:06.9162	12:29.4192	13.0879	1	75
NonStrictInit+NonStrictBM+BestDevLS	12:06.9760	12:06.9162	12:06.9162	12:21.8473	13.4902	1	95
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	12:06.9760	12:17.0060	12:06.9760	12:25.5928	9.6906	1	65
NonStrictInit+NonStrictBM-SA+IndepLS	12:14.2814	12:15.7485	12:14.2814	12:24.0958	8.9017	1	95
SA	12:24.3713	12:15.0599	12:15.0599	12:23.7844	6.5765	1	97
StrictSA	–	12:49.2216	12:49.2216	12:49.2216	0.0000	10	54
StrictInit+StrictBM-SA+StrictBestDevLS	–	12:49.2216	12:49.2216	12:50.3353	1.7282	5	53
StrictInit+StrictBM-SA+StrictIndepLS-SA	–	12:49.2216	12:49.2216	12:50.5868	1.8252	3	59
StrictInit+StrictBM+StrictBestDevLS	–	12:49.2216	12:49.2216	12:50.3114	1.8923	6	24
StrictInit+StrictBM-SA+StrictBestDevLS-SA	–	12:49.2216	12:49.2216	12:51.0180	1.9703	2	29
StrictInit+StrictBM+StrictBestDevLS-SA	–	12:49.2216	12:49.2216	12:50.8263	2.2328	5	41
StrictInit+StrictBM+StrictIndepLS-SA	–	12:49.2216	12:49.2216	12:51.6886	2.4532	4	18
StrictInit+StrictBM+StrictContLS	–	12:49.2216	12:49.2216	12:51.0419	2.4961	4	53
StrictInit+StrictBM-SA+StrictIndepLS	–	12:49.2216	12:49.2216	12:51.2216	3.0741	5	59
StrictInit+StrictBM-SA+StrictContLS	–	12:49.2216	12:49.2216	12:51.6527	3.7432	6	18
StrictInit+StrictBM+StrictIndepLS	–	12:49.2216	12:49.2216	12:51.6527	4.1448	7	53
StrictInit+StrictBM+StrictContLS-SA	–	12:49.2216	12:49.2216	12:52.6108	4.5066	4	71
StrictInit+StrictBM-SA+StrictContLS-SA	–	12:49.2216	12:49.2216	12:53.3174	5.0969	2	71

Table 25: Results of O10 optimisation - sequenced by standard deviation

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+StrictBM+StrictBestDevLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictBestDevLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictContLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictContLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictIndepLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictIndepLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictBestDevLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictBestDevLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictContLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictContLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictIndepLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictIndepLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictSA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	1
StrictInit+NonStrictBM+ContLS-SA	–	7:04.6707	7:04.6707	7:05.9162	2.6258	8	20
StrictInit+NonStrictBM-SA+BestDevLS-SA	–	7:04.6707	7:04.6707	7:06.3323	3.6379	8	60
StrictInit+NonStrictBM-SA+IndepLS	–	7:04.6707	7:04.6707	7:06.3323	3.6379	8	60
StrictInit+NonStrictBM+BestDevLS-SA	–	7:04.6707	7:04.6707	7:06.0778	4.4499	9	40
SA	7:20.3593	7:04.6707	7:04.6707	7:07.5329	4.8182	7	59
NonStrictInit+NonStrictBM+IndepLS-SA	7:10.8982	7:04.6707	7:04.6707	7:09.8174	4.8474	4	60
NonStrictInit+NonStrictBM-SA+IndepLS-SA	7:24.3114	7:04.6707	7:04.6707	7:10.2335	5.1214	4	60
NonStrictInit+NonStrictBM+ContLS-SA	7:10.8982	7:04.6707	7:04.6707	7:11.4790	5.2045	3	60
StrictInit+NonStrictBM-SA+IndepLS-SA	–	7:04.6707	7:04.6707	7:08.7335	5.4766	6	60
StrictInit+NonStrictBM-SA+ContLS-SA	–	7:04.6707	7:04.6707	7:09.1946	5.9364	6	20
NonStrictInit+NonStrictBM-SA+BestDevLS	7:04.6707	7:04.6707	7:04.6707	7:10.1407	6.0901	5	40
NonStrictInit+NonStrictBM+BestDevLS-SA	7:42.6048	7:04.6707	7:04.6707	7:13.0958	6.6525	3	60
StrictInit+NonStrictBM-SA+ContLS	–	7:04.6707	7:04.6707	7:09.0329	6.6549	6	20
StrictInit+NonStrictBM-SA+BestDevLS	–	7:04.6707	7:04.6707	7:11.9910	6.8781	4	80
StrictInit+NonStrictBM+IndepLS-SA	–	7:04.6707	7:04.6707	7:08.0419	7.2274	8	20
NonStrictInit+NonStrictBM+ContLS	7:04.6707	7:04.6707	7:04.6707	7:09.7036	7.2816	6	40
StrictInit+NonStrictBM+BestDevLS	–	7:04.6707	7:04.6707	7:08.9192	7.6023	7	20
NonStrictInit+NonStrictBM-SA+ContLS-SA	7:26.6168	7:04.6707	7:04.6707	7:12.3084	7.6995	4	60
NonStrictInit+NonStrictBM-SA+ContLS	7:32.8443	7:04.6707	7:04.6707	7:09.5629	7.9720	7	40
NonStrictInit+NonStrictBM-SA+IndepLS	7:26.6168	7:04.6707	7:04.6707	7:10.1407	8.1321	6	60
StrictInit+NonStrictBM+ContLS	–	7:04.6707	7:04.6707	7:10.4222	8.3243	6	40
StrictInit+NonStrictBM+IndepLS	–	7:04.6707	7:04.6707	7:13.0749	8.6491	4	40
NonStrictInit+NonStrictBM+BestDevLS	7:32.8443	7:04.6707	7:04.6707	7:13.2335	9.2154	5	60
NonStrictInit+NonStrictBM+IndepLS	7:24.3114	7:04.6707	7:04.6707	7:12.4371	9.3769	5	40
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	7:22.9042	7:04.6707	7:04.6707	7:14.9910	10.7882	4	60

Table 26: Results of O20 optimisation - sequenced by standard deviation

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictSA	–	12:49.2216	12:49.2216	12:49.2216	0.0000	10	54
StrictInit+StrictBM-SA+StrictBestDevLS	–	12:49.2216	12:49.2216	12:50.3353	1.7282	5	53
StrictInit+StrictBM-SA+StrictIndepLS-SA	–	12:49.2216	12:49.2216	12:50.5868	1.8252	3	59
StrictInit+StrictBM+StrictBestDevLS	–	12:49.2216	12:49.2216	12:50.3114	1.8923	6	24
StrictInit+StrictBM-SA+StrictBestDevLS-SA	–	12:49.2216	12:49.2216	12:51.0180	1.9703	2	29
StrictInit+StrictBM+StrictBestDevLS-SA	–	12:49.2216	12:49.2216	12:50.8263	2.2328	5	41
StrictInit+StrictBM+StrictIndepLS-SA	–	12:49.2216	12:49.2216	12:51.6886	2.4532	4	18
StrictInit+StrictBM+StrictContLS	–	12:49.2216	12:49.2216	12:51.0419	2.4961	4	53
StrictInit+StrictBM-SA+StrictIndepLS	–	12:49.2216	12:49.2216	12:51.2216	3.0741	5	59
StrictInit+StrictBM-SA+StrictContLS	–	12:49.2216	12:49.2216	12:51.6527	3.7432	6	18
StrictInit+StrictBM+StrictIndepLS	–	12:49.2216	12:49.2216	12:51.6527	4.1448	7	53
StrictInit+StrictBM+StrictContLS-SA	–	12:49.2216	12:49.2216	12:52.6108	4.5066	4	71
StrictInit+StrictBM-SA+StrictContLS-SA	–	12:49.2216	12:49.2216	12:53.3174	5.0969	2	71
SA	12:24.3713	12:15.0599	12:15.0599	12:23.7844	6.5765	1	97
StrictInit+NonStrictBM+BestDevLS-SA	–	11:56.5868	11:56.5868	12:08.6228	8.3947	1	100
NonStrictInit+NonStrictBM-SA+IndepLS	12:14.2814	12:15.7485	12:14.2814	12:24.0958	8.9017	1	95
StrictInit+NonStrictBM+ContLS-SA	–	11:56.1377	11:56.1377	12:13.3204	9.0991	1	85
StrictInit+NonStrictBM-SA+ContLS	–	11:53.5030	11:53.5030	12:07.7695	9.4708	1	95
StrictInit+NonStrictBM-SA+IndepLS	–	12:04.9102	12:04.9102	12:17.6497	9.6629	1	95
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	12:06.9760	12:17.0060	12:06.9760	12:25.5928	9.6906	1	65
StrictInit+NonStrictBM-SA+BestDevLS	–	11:58.5928	11:58.5928	12:18.4042	10.2987	1	100
StrictInit+NonStrictBM+BestDevLS	–	11:53.4431	11:53.4431	12:11.0210	10.7707	1	80
StrictInit+NonStrictBM-SA+BestDevLS-SA	–	11:56.9760	11:56.9760	12:17.2904	10.7730	1	80
StrictInit+NonStrictBM+IndepLS	–	11:51.1976	11:51.1976	12:11.9731	10.8084	1	100
StrictInit+NonStrictBM-SA+ContLS-SA	–	11:55.2695	11:55.2695	12:12.8263	11.8230	1	55
NonStrictInit+NonStrictBM+BestDevLS-SA	12:02.3054	11:58.4431	11:58.4431	12:14.2006	11.9146	1	100
NonStrictInit+NonStrictBM+IndepLS	11:51.3473	12:09.0419	11:51.3473	12:17.0599	12.0535	1	90
NonStrictInit+NonStrictBM+ContLS	12:17.0659	11:58.4431	11:58.4431	12:19.1317	12.8389	1	80
NonStrictInit+NonStrictBM-SA+IndepLS-SA	12:31.4671	12:06.9162	12:06.9162	12:29.4192	13.0879	1	75
NonStrictInit+NonStrictBM+BestDevLS	12:06.9760	12:06.9162	12:06.9162	12:21.8473	13.4902	1	95
StrictInit+NonStrictBM+ContLS	–	11:51.3473	11:51.3473	12:09.7784	13.9960	1	85
NonStrictInit+NonStrictBM-SA+ContLS-SA	11:58.5928	12:19.8204	11:58.5928	12:27.5928	14.1030	1	50
StrictInit+NonStrictBM-SA+IndepLS-SA	–	11:58.3533	11:58.3533	12:17.7335	14.4560	1	90
StrictInit+NonStrictBM+IndepLS-SA	–	11:51.3473	11:51.3473	12:10.4671	14.4946	1	80
NonStrictInit+NonStrictBM-SA+BestDevLS	12:04.8204	11:53.5030	11:53.5030	12:17.0269	14.7494	1	90
NonStrictInit+NonStrictBM-SA+ContLS	12:22.4251	11:56.5868	11:56.5868	12:21.1886	14.8191	1	95
NonStrictInit+NonStrictBM+IndepLS-SA	12:12.1257	11:53.4431	11:53.4431	12:18.5329	15.2699	1	80
NonStrictInit+NonStrictBM+ContLS-SA	11:51.1976	11:56.1377	11:51.1976	12:20.4970	20.4119	1	65

the strict alternatives for both of the examined order picking lists. It could be said, that the allowance of reconstruction could result in lower lead time.

BestEugLT and BestNotEugLT columns prove that the eugenic bacterium rarely gave the BestLT of the algorithms. However it might help the improvement indirectly via the GT operator.

The high BestOcc values of Table 27 highlight, that the algorithms find a nearly optimal solution for the O10 list. However, Table 28 shows that the non-strict algorithms reached the BestLT only once in the case of O20. The 1stBestGen% values of the O20 case are also higher than in the O10 table. BestOcc and 1stBestGen% also highlight, that O20 algorithms might need more budget to reach near optimal results. However, it will increase the necessary optimisation time.

The results show that several non-strict BMA algorithms performed better than an SA algorithm on the same budget in AVG and from the BestLT point of view. It can be said, that several proposed BMA algorithms are more effective than the SA algorithm; a fact of which highlights their effectiveness.

Table 29 and Table 30 summarise the best alternatives of the initial population defining method and BM operator combinations from an AVG point of view in increasing order of AVG. These tables show that the strict initial population performed well with non-strict BM operators. The fully non-strict algorithms are in the mid-range and the fully strict algorithms are at the end of the list. The best algorithm of O20 applied a non-strict BM with SA method. The NonStrictInit+NonStrictBM-SA+ContLS algorithm performed better than NonStrictInit+NonStrictBM+ContLS in the case of O10. These results confirm the operability of using SA in a BM operator. The BestDevLS and BestDevLS-SA LS operators could perform well in the case of longer picking lists because most of the algorithms used these for O20.

Figure 17 depicts the iterations of the examined algorithms for O20. The algorithms, which used SA, combined with the BM operator performed with slower improvement in the first half of the iteration than the algorithms with the same initial population method and BM operator without SA. These characteristics are caused by the nature of SA. However, finally, BM operators combined with SA competed with the simple BM operators. Figure 18 visualises the nature of SA algorithms for O20, which has a slower improvement than the BMA algorithms. It highlights, that the strict SA algorithm has already found its optimum, when the non-strict algorithm is still continuously improving the picking sequence.

Table 31 summarises the occurrence of the possible InitPop and BM operator combinations within the AVG based top10 BMA algorithms. It also highlights, that the strict initial population and non-strict BM are the most frequent combination with and without SA for both O10 and O20 cases. The occurrence of a SA combined operator is decreased in O20 case, which highlights the decreasing efficiency of SA combined operators in the

Table 27: Results of O10 optimisation - sequenced by average lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM+ContLS-SA	–	7:04.6707	7:04.6707	7:05.9162	2.6258	8	20
StrictInit+NonStrictBM+BestDevLS-SA	–	7:04.6707	7:04.6707	7:06.0778	4.4499	9	40
StrictInit+NonStrictBM-SA+BestDevLS-SA	–	7:04.6707	7:04.6707	7:06.3323	3.6379	8	60
StrictInit+NonStrictBM-SA+IndepLS	–	7:04.6707	7:04.6707	7:06.3323	3.6379	8	60
SA	7:20.3593	7:04.6707	7:04.6707	7:07.5329	4.8182	7	59
StrictInit+NonStrictBM+IndepLS-SA	–	7:04.6707	7:04.6707	7:08.0419	7.2274	8	20
StrictInit+NonStrictBM-SA+IndepLS-SA	–	7:04.6707	7:04.6707	7:08.7335	5.4766	6	60
StrictInit+NonStrictBM+BestDevLS	–	7:04.6707	7:04.6707	7:08.9192	7.6023	7	20
StrictInit+NonStrictBM-SA+ContLS	–	7:04.6707	7:04.6707	7:09.0329	6.6549	6	20
StrictInit+NonStrictBM-SA+ContLS-SA	–	7:04.6707	7:04.6707	7:09.1946	5.9364	6	20
NonStrictInit+NonStrictBM-SA+ContLS	7:32.8443	7:04.6707	7:04.6707	7:09.5629	7.9720	7	40
NonStrictInit+NonStrictBM+ContLS	7:04.6707	7:04.6707	7:04.6707	7:09.7036	7.2816	6	40
NonStrictInit+NonStrictBM+IndepLS-SA	7:10.8982	7:04.6707	7:04.6707	7:09.8174	4.8474	4	60
NonStrictInit+NonStrictBM-SA+BestDevLS	7:04.6707	7:04.6707	7:04.6707	7:10.1407	6.0901	5	40
NonStrictInit+NonStrictBM-SA+IndepLS	7:26.6168	7:04.6707	7:04.6707	7:10.1407	8.1321	6	60
NonStrictInit+NonStrictBM-SA+IndepLS-SA	7:24.3114	7:04.6707	7:04.6707	7:10.2335	5.1214	4	60
StrictInit+NonStrictBM+ContLS	–	7:04.6707	7:04.6707	7:10.4222	8.3243	6	40
NonStrictInit+NonStrictBM+ContLS-SA	7:10.8982	7:04.6707	7:04.6707	7:11.4790	5.2045	3	60
StrictInit+NonStrictBM-SA+BestDevLS	–	7:04.6707	7:04.6707	7:11.9910	6.8781	4	80
NonStrictInit+NonStrictBM-SA+ContLS-SA	7:26.6168	7:04.6707	7:04.6707	7:12.3084	7.6995	4	60
NonStrictInit+NonStrictBM+IndepLS	7:24.3114	7:04.6707	7:04.6707	7:12.4371	9.3769	5	40
StrictInit+NonStrictBM+IndepLS	–	7:04.6707	7:04.6707	7:13.0749	8.6491	4	40
NonStrictInit+NonStrictBM+BestDevLS-SA	7:42.6048	7:04.6707	7:04.6707	7:13.0958	6.6525	3	60
NonStrictInit+NonStrictBM+BestDevLS	7:32.8443	7:04.6707	7:04.6707	7:13.2335	9.2154	5	60
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	7:22.9042	7:04.6707	7:04.6707	7:14.9910	10.7882	4	60
StrictInit+StrictBM+StrictBestDevLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictBestDevLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictContLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictContLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictIndepLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM+StrictIndepLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictBestDevLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictBestDevLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictContLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictContLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictIndepLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictIndepLS-SA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictSA	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	1

Table 28: Results of O20 optimisation - sequenced by average lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM-SA+ContLS	–	11:53.5030	11:53.5030	12:07.7695	9.4708	1	95
StrictInit+NonStrictBM+BestDevLS-SA	–	11:56.5868	11:56.5868	12:08.6228	8.3947	1	100
StrictInit+NonStrictBM+ContLS	–	11:51.3473	11:51.3473	12:09.7784	13.9960	1	85
StrictInit+NonStrictBM+IndepLS-SA	–	11:51.3473	11:51.3473	12:10.4671	14.4946	1	80
StrictInit+NonStrictBM+BestDevLS	–	11:53.4431	11:53.4431	12:11.0210	10.7707	1	80
StrictInit+NonStrictBM+IndepLS	–	11:51.1976	11:51.1976	12:11.9731	10.8084	1	100
StrictInit+NonStrictBM-SA+ContLS-SA	–	11:55.2695	11:55.2695	12:12.8263	11.8230	1	55
StrictInit+NonStrictBM+ContLS-SA	–	11:56.1377	11:56.1377	12:13.3204	9.0991	1	85
NonStrictInit+NonStrictBM+BestDevLS-SA	12:02.3054	11:58.4431	11:58.4431	12:14.2006	11.9146	1	100
NonStrictInit+NonStrictBM-SA+BestDevLS	12:04.8204	11:53.5030	11:53.5030	12:17.0269	14.7494	1	90
NonStrictInit+NonStrictBM+IndepLS	11:51.3473	12:09.0419	11:51.3473	12:17.0599	12.0535	1	90
StrictInit+NonStrictBM-SA+BestDevLS-SA	–	11:56.9760	11:56.9760	12:17.2904	10.7730	1	80
StrictInit+NonStrictBM-SA+IndepLS	–	12:04.9102	12:04.9102	12:17.6497	9.6629	1	95
StrictInit+NonStrictBM-SA+IndepLS-SA	–	11:58.3533	11:58.3533	12:17.7335	14.4560	1	90
StrictInit+NonStrictBM-SA+BestDevLS	–	11:58.5928	11:58.5928	12:18.4042	10.2987	1	100
NonStrictInit+NonStrictBM+IndepLS-SA	12:12.1257	11:53.4431	11:53.4431	12:18.5329	15.2699	1	80
NonStrictInit+NonStrictBM+ContLS	12:17.0659	11:58.4431	11:58.4431	12:19.1317	12.8389	1	80
NonStrictInit+NonStrictBM+ContLS-SA	11:51.1976	11:56.1377	11:51.1976	12:20.4970	20.4119	1	65
NonStrictInit+NonStrictBM-SA+ContLS	12:22.4251	11:56.5868	11:56.5868	12:21.1886	14.8191	1	95
NonStrictInit+NonStrictBM+BestDevLS	12:06.9760	12:06.9162	12:06.9162	12:21.8473	13.4902	1	95
SA	12:24.3713	12:15.0599	12:15.0599	12:23.7844	6.5765	1	97
NonStrictInit+NonStrictBM-SA+IndepLS	12:14.2814	12:15.7485	12:14.2814	12:24.0958	8.9017	1	95
NonStrictInit+NonStrictBM-SA+BestDevLS-SA	12:06.9760	12:17.0060	12:06.9760	12:25.5928	9.6906	1	65
NonStrictInit+NonStrictBM-SA+ContLS-SA	11:58.5928	12:19.8204	11:58.5928	12:27.5928	14.1030	1	50
NonStrictInit+NonStrictBM-SA+IndepLS-SA	12:31.4671	12:06.9162	12:06.9162	12:29.4192	13.0879	1	75
StrictSA	–	12:49.2216	12:49.2216	12:49.2216	0.0000	10	54
StrictInit+StrictBM+StrictBestDevLS	–	12:49.2216	12:49.2216	12:50.3114	1.8923	6	24
StrictInit+StrictBM-SA+StrictBestDevLS	–	12:49.2216	12:49.2216	12:50.3353	1.7282	5	53
StrictInit+StrictBM-SA+StrictIndepLS-SA	–	12:49.2216	12:49.2216	12:50.5868	1.8252	3	59
StrictInit+StrictBM+StrictBestDevLS-SA	–	12:49.2216	12:49.2216	12:50.8263	2.2328	5	41
StrictInit+StrictBM-SA+StrictBestDevLS-SA	–	12:49.2216	12:49.2216	12:51.0180	1.9703	2	29
StrictInit+StrictBM+StrictContLS	–	12:49.2216	12:49.2216	12:51.0419	2.4961	4	53
StrictInit+StrictBM-SA+StrictIndepLS	–	12:49.2216	12:49.2216	12:51.2216	3.0741	5	59
StrictInit+StrictBM-SA+StrictContLS	–	12:49.2216	12:49.2216	12:51.6527	3.7432	6	18
StrictInit+StrictBM+StrictIndepLS	–	12:49.2216	12:49.2216	12:51.6527	4.1448	7	53
StrictInit+StrictBM+StrictIndepLS-SA	–	12:49.2216	12:49.2216	12:51.6886	2.4532	4	18
StrictInit+StrictBM+StrictContLS-SA	–	12:49.2216	12:49.2216	12:52.6108	4.5066	4	71
StrictInit+StrictBM-SA+StrictContLS-SA	–	12:49.2216	12:49.2216	12:53.3174	5.0969	2	71

case of long lists.

Figure 17: The best performing combinations of Initial populations and BM operators for O20

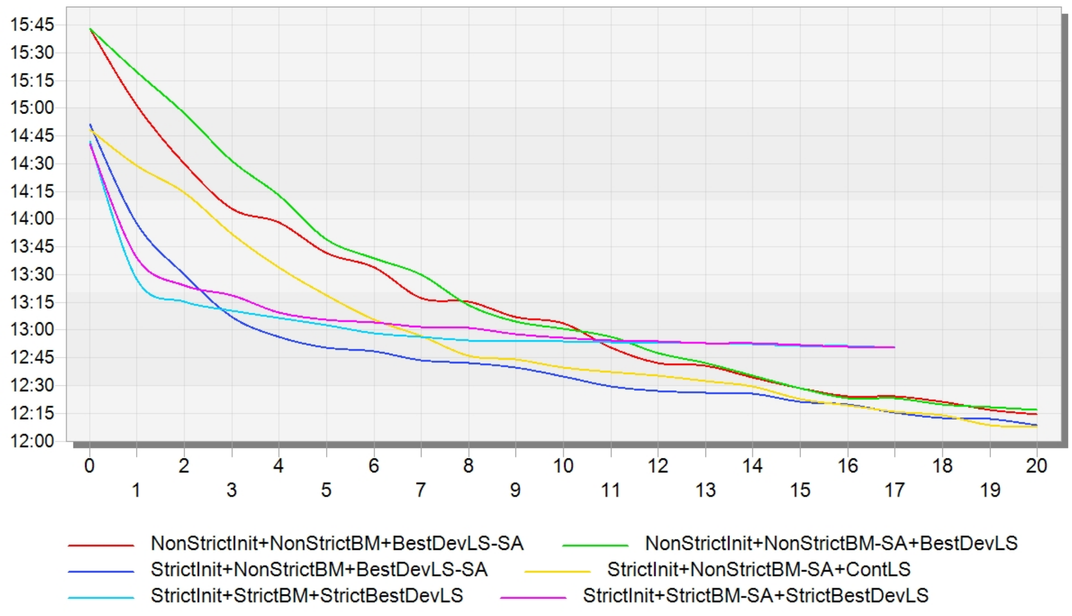


Table 29 and Table 30 summarise the best alternatives of each LS operators from an AVG point of view in increasing order of AVG. The ContLS operators are used in the best and worst alternatives. The BestDevLS operator is situated in the strong mid-range. The IndepLS operators are in the mid-range but are usually weaker than any of BestDevLS operators.

The LS operators are combined with the SA several times to get the best results compared to those without the SA alternative, which confirm its operability. In the case of O20 the BestDevLS operators get the best of IndepLS operators in the SA completed version compared to the version without SA. However the IndepLS-SA operator performs better than the simple BestDevLS operator. It can be said that the BestDevLS operators, mainly the BestDevLS-SA operator, perform stably and effectively for both short and long lists. Furthermore, each of the best LS algorithms used the StrictInit initial population and one of the non-strict BM operators, which proves the effectiveness of this combination. Figure 19 shows the iterations of the examined algorithms for O20.

The occurrence of LS operators within the AVG based top10 BMA algorithms has been summarised in Table 34 for both lists. It highlights, that while the IndepLS and IndepLS-SA operators' productivity is decreased for O20, the further operators' occurrence is equal and stable.

Table 35 examines the occurrence of BMA operators combined with SA within the AVG based top 10 BMA algorithms. It highlights, that in the case of O10 90% of the top 10 algorithms used an SA combined operator, and in the case of O20 the occurrence of non-SA combined operators is increased. In the case of longer (O20) lists the algorithms, which

Table 29: Best algorithms of Initial population and BM operator combinations during O10 optimisation - sequenced by average lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM+ContLS-SA	–	7:04.6707	7:04.6707	7:05.9162	2.6258	8	20
StrictInit+NonStrictBM-SA+BestDevLS-SA	–	7:04.6707	7:04.6707	7:06.3323	3.6379	8	60
NonStrictInit+NonStrictBM-SA+ContLS	7:32.8443	7:04.6707	7:04.6707	7:09.5629	7.9720	7	40
NonStrictInit+NonStrictBM+ContLS	7:04.6707	7:04.6707	7:04.6707	7:09.7036	7.2816	6	40
StrictInit+StrictBM+StrictBestDevLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25
StrictInit+StrictBM-SA+StrictBestDevLS	–	7:24.3114	7:24.3114	7:24.3114	0.0000	10	25

Table 30: Best algorithms of Initial population and BM operator combinations during O20 optimisation - sequenced by average lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM-SA+ContLS	–	11:53.5030	11:53.5030	12:07.7695	9.4708	1	95
StrictInit+NonStrictBM+BestDevLS-SA	–	11:56.5868	11:56.5868	12:08.6228	8.3947	1	100
NonStrictInit+NonStrictBM+BestDevLS-SA	12:02.3054	11:58.4431	11:58.4431	12:14.2006	11.9146	1	100
NonStrictInit+NonStrictBM-SA+BestDevLS	12:04.8204	11:53.5030	11:53.5030	12:17.0269	14.7494	1	90
StrictInit+StrictBM+StrictBestDevLS	–	12:49.2216	12:49.2216	12:50.3114	1.8923	6	24
StrictInit+StrictBM-SA+StrictBestDevLS	–	12:49.2216	12:49.2216	12:50.3353	1.7282	5	53

Table 31: Occurrence of InitPop and BM operator combinations within top 10 BMA algorithms based on Avg

InitPop-BM combination	O10	O20
NonStrictInit+NonStrictBM	0	1
NonStrictInit+NonStrictBM-SA	1	1
StrictInit+NonStrictBM	4	6
StrictInit+NonStrictBM-SA	5	2
StrictInit+StrictBM	0	0
StrictInit+StrictBM-SA	0	0

Figure 18: The best performing combinations of Initial population and BM operators for O20

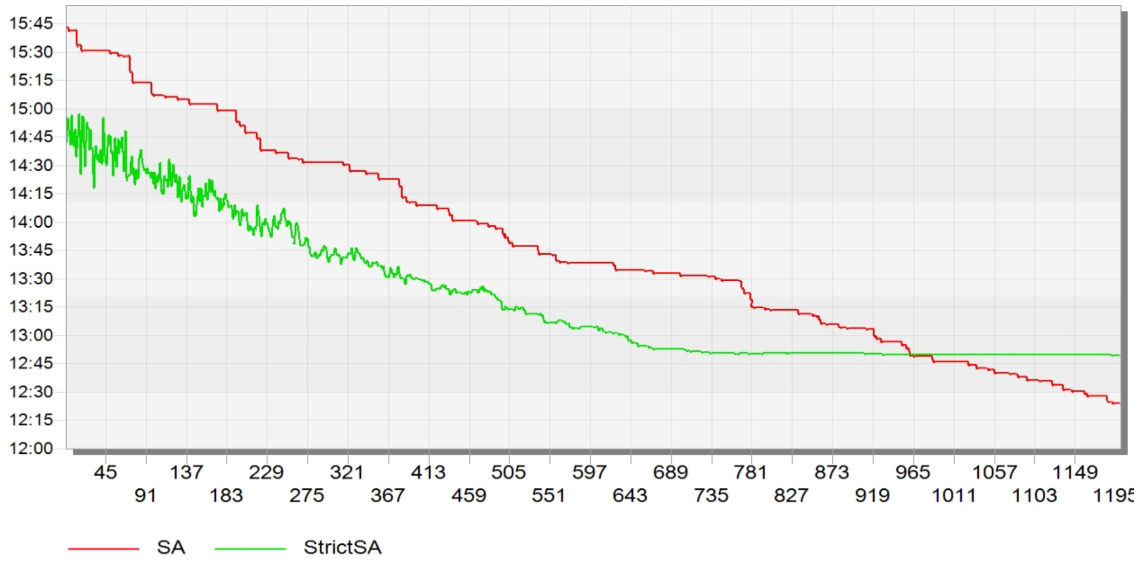


Figure 19: The best performing combinations of LS operators

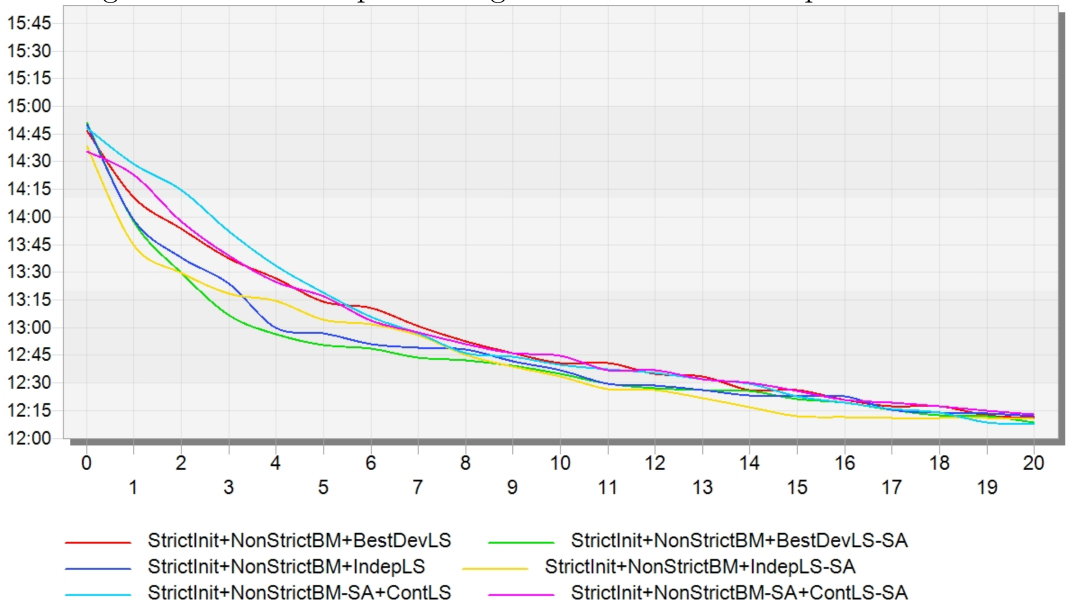


Table 32: Best algorithms of LS operators during O10 optimisation - sequenced by average lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM+ContLS-SA	–	7:04.6707	7:04.6707	7:05.9162	2.6258	8	20
StrictInit+NonStrictBM+BestDevLS-SA	–	7:04.6707	7:04.6707	7:06.0778	4.4499	9	40
StrictInit+NonStrictBM-SA+IndepLS	–	7:04.6707	7:04.6707	7:06.3323	3.6379	8	60
StrictInit+NonStrictBM+IndepLS-SA	–	7:04.6707	7:04.6707	7:08.0419	7.2274	8	20
StrictInit+NonStrictBM+BestDevLS	–	7:04.6707	7:04.6707	7:08.9192	7.6023	7	20
StrictInit+NonStrictBM-SA+ContLS	–	7:04.6707	7:04.6707	7:09.0329	6.6549	6	20

Table 33: Best algorithms of LS operators during O20 optimisation - sequenced by average lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM-SA+ContLS	–	11:53.5030	11:53.5030	12:07.7695	9.4708	1	95
StrictInit+NonStrictBM+BestDevLS-SA	–	11:56.5868	11:56.5868	12:08.6228	8.3947	1	100
StrictInit+NonStrictBM+IndepLS-SA	–	11:51.3473	11:51.3473	12:10.4671	14.4946	1	80
StrictInit+NonStrictBM+BestDevLS	–	11:53.4431	11:53.4431	12:11.0210	10.7707	1	80
StrictInit+NonStrictBM+IndepLS	–	11:51.1976	11:51.1976	12:11.9731	10.8084	1	100
StrictInit+NonStrictBM-SA+ContLS-SA	–	11:55.2695	11:55.2695	12:12.8263	11.8230	1	55

used BM operators combined with SA lost their effectiveness. However, the algorithms, which used a BM operator without SA and an LS operator combined with SA have increased occurrence in the case of longer lists (O20). It confirms the productivity of those algorithms, which combine an LS operator only with SA.

Nine non-strict BMA and the non-strict SA algorithms were used for the O50 order picking list, whose complexity is described in Table 21. The selected algorithms performed well in the previous lists. Table 36 shows the O50 results sequenced by AVG. The strict initialisation and the non-strict BM operators reached the lowest average lead time without any SA.

Each BMA algorithm performed better than the SA algorithm, which strongly proves the effectiveness of BMA algorithms for OPRP-PLF. However, the results show the SA completed algorithms lose to the BMA operators without SA in the case of extremely complex lists. It can be said that it is productive to use SA combined operators in the case of less complex picking tasks (for example O10 and O20), when these performed well.

The simulation results highlighted the following consequences:

- While the reconstruction time is less than the saved travelling time, allowance of reconstruction could result in lower order picking lead time.
- Most of the well-performing algorithms used a strict initial population with one of the non-strict bacterial mutations. It highlighted, that non-strict operators are necessary to allow reconstruction but a strict initial population helps with quick improvement.
- BestDevLS operators, mainly the BestDevLS-SA operator, performs stably and effectively for both short and long lists.
- Comparing the non-strict SA algorithm with non-strict BMA algorithms, several BMA operator combinations performed better than the SA in AVG and BestLT with the same budget. Several BMA algorithms are more effective than a SA algorithm.
- The SA combined bacterial mutation and local search operators competed with

Table 34: Occurrence of LS operators within top 10 BMA algorithms based on Avg

LS alternatives	O10	O20
BestDevLS	1	2
BestDevLS-SA	2	2
ContLS	2	2
ContLS-SA	2	2
IndepLS	1	1
IndepLS-SA	2	1

Table 35: Occurrence of SA combined operators within the top 10 BMA algorithms based on Avg

SA combined operator possibilities	O10	O20
BM-SA	3	2
LS-SA	3	4
BM-SA + LS-SA	3	1
Non-SA combined operator	1	3

simple operators, which confirm the operability of applying SA into BMA operators. Mainly the SA combined LS operators get the best of its alternatives without SA.

- The effectiveness of a SA combined operator is decreased in the O20 and O50 cases, which highlights the decreasing efficiency of SA combined operators in the case of long lists. However, the number of records for one unit load are usually under 20 in practice when SA combined algorithms performed well.

Finally I can conclude that BMA algorithms are more effective for the proposed OPRP-PLF than the SA algorithm. For further research, the StrictInit+NonStrictBM+BestDevLS-SA algorithm will be preferred because of its stable results. To support the possible reproduction of the solution the Appendix Section shows the optimised sequence of the applied order picking lists. These sequences resulted in the lowest LeadTime of the selected algorithm.

Table 36: Results of O50 optimisation - sequenced by average lead time

Algorithm	BestEugLT	BestNotEugLT	BestLT	AVG	SD	BestOcc	1stBestGen%
StrictInit+NonStrictBM+BestDevLS	–	25:58.5928	25:58.5928	26:51.6018	30.4412	1	100
StrictInit+NonStrictBM+ContLS	–	26:23.9820	26:23.9820	26:54.9401	14.8389	1	63
StrictInit+NonStrictBM-SA+BestDevLS-SA	–	26:13.4731	26:13.4731	27:00.0000	26.8709	1	75
StrictInit+NonStrictBM-SA+ContLS-SA	–	26:26.7066	26:26.7066	27:00.9072	29.6944	1	99
StrictInit+NonStrictBM+BestDevLS-SA	–	26:24.9102	26:24.9102	27:01.1737	24.1018	1	62
StrictInit+NonStrictBM+IndepLS-SA	–	26:16.5868	26:16.5868	27:01.4371	27.7974	1	83
NonStrictInit+NonStrictBM+BestDevLS-SA	27:49.4611	26:16.2874	26:16.2874	27:08.3892	32.8531	1	84
StrictInit+NonStrictBM-SA+ContLS	–	26:45.0898	26:45.0898	27:09.3533	25.2259	1	98
NonStrictInit+NonStrictBM-SA+BestDevLS	27:35.0898	26:30.8982	26:30.8982	27:10.0539	25.1130	1	95
SA	28:29.7904	27:48.6826	27:48.6826	28:41.1946	37.6553	1	100

3.4.4 Summarising statements

Evaluation of several possible algorithm is usually necessary to define the right concept for a novel problem. I defined possible BMA solutions for OPRP-PLF optimisation based on the possible combinations of my proposed BMA operators. I implemented, evaluated, and compared each BMA and the SA algorithms on the same basis with my computer simulation model. I examined the behaviour of the algorithms on order picking lists with different record numbers.

Thesis statement 4. *Based on the results I proved, that BMA is more effective for the proposed OPRP-PLF than the population based SA algorithm. I concluded, that for most of the applications the strictly initialised, non-strict bacterial mutation, best development local search combined with SA algorithm could be an effective choice.*

Thesis statement 4/a. *I verified, that when the reconstruction time is less than the saved travelling time, then the allowance of reconstruction could result in a lower order picking lead time. However, the non-strict operators are necessary to allow reconstruction but a strict initial population helps with quick improvement.*

Thesis statement 4/b. *I proved, that the Best Development Local Search operators, mainly its SA combined version, perform stably and effectively for both short and long lists.*

Thesis statement 4/c. *I highlighted the operability of applying SA into BMA operators, since the SA combined bacterial mutation and local search operators are competitive with the traditional operators. Mainly the SA combined Local Search operators obtained better results than their alternatives without SA.*

My publication related to the statement: [9].

Table 37: Pallet Loading Feature based Decision Matrix (PLFDM) of the case study

	Bag	*Pail*	*Can*	*BigBox*	*CanLQ*	*SmallBox*	*Fragile*
Bag	1	1	1	1	1	1	1
Pail	0	1	1	1	1	1	1
Can	0	0	1	1	1	1	1
BigBox	0	0	0	1	1	1	1
CanLQ	0	0	0	0	1	1	1
SmallBox	0	0	0	0	0	1	1
Fragile	0	0	0	0	0	0	1

3.5 The warehouse layout and the SLA effect on the OPRP-PLF

The proposed Plant Simulation model makes me able to configure several system attributes to evaluate them on an objective way. The combination of the different system attributes defines alternatives for evaluation, which will be done based on order picking of different characteristics orders.

The aim of this section is to examine the effect of the warehouse layout attributes, the SLA and the UL reconstruction on the order picking lead time.

I determined industrially relevant alternatives for the mentioned system attributes to evaluate them on several different characteristics orders. Based on the objective evaluation of alternative system configuration I am able to make conclusions about the contexts of the parameters and necessity of solutions.

3.5.1 System attributes of the case study

The general attributes (e.g. time parameters, speed) and the behaviour of the model are the same as discussed before (Section 3.4.) during routing algorithm development.

The applied PLFDM (Table 37) contains a Low Quantity Can (*CanLQ*) POPC which is applied when the ordered quantity is less than one layer of products.

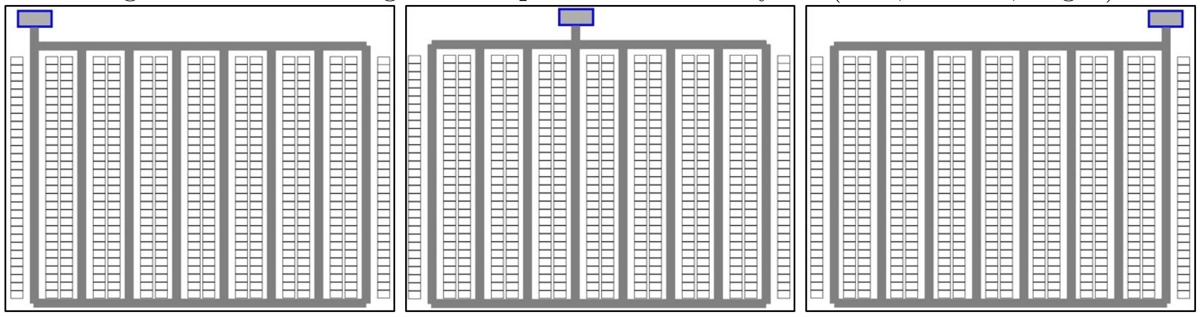
3.5.1.1 Layout attributes

I assumed based on literature reviews and industrial experiences, that the layout geometric dimensions and the positioning of the Departure and Arrival (DA) position have an impact on the order picking lead time in the case of relevant PLF [8].

Therefore I defined "Small" and "Big" layout alternatives, where the "Small" layout applies realistic distances between the 480 implemented order picking positions. It results in approximately 0.9 m distances between the mid-point of the neighbouring order picking positions. The "Big" layout applies doubled distances for the corridors and the cross-lines, which results in four times bigger layout [8].

Three alternatives have been defined for the DA position, when it is on the left, middle or right side of the manipulation area (Fig. 20.). Examining of these alternatives is important, because the departure position of the picking can be different order by order based on the forwarding area/gate. The effectiveness of a SLA is influenced where the picker starts and finishes the route. It can be more important, when the SLA is defined based on theoretical picking sequence of the PCes [8].

Figure 20: Positioning the DA position on the layout (Left, Middle, Right)



I defined the previously described layout alternatives as follows:

- Layout dimensions
  - "Small" layout
  - "Big" layout
- Positioning of the Departure and Arrival position
  - Left
  - Middle
  - Right

3.5.1.2 Storage Location Assignment

The main aim of SLA is to support the routing algorithm minimising the order picking lead time and costs on the given layout [21]. Therefore, the PLFs should be considered comprehensively, because both the product parameters, the order characteristics, and the order picking system itself have an impact on the order picking lead time and cost.

I defined two SLA cases to evaluate those effects on the order picking lead time. The first one is a totally randomised SLA, when the 480 products are allocated to picking position without any rule. The second one is a PLF based SLA, when I defined the theoretical sequence (Table 38) of the PCes and the 480 products are allocated to picking position based on this sequence PC by PC from left to right. The allocation of the products are randomised within the PC zones (Fig. 21) [8].

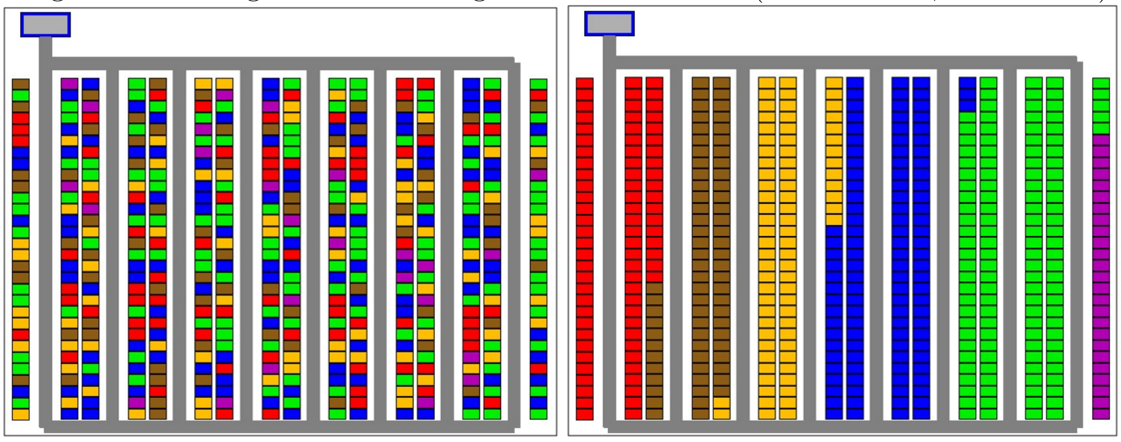
I defined the previously described SLA alternatives as follows:

- Randomised (RND) SLA
- PLF based SLA

Table 38: Legend and theoretical picking sequence

Theoretical stacking sequence	Stacking classes
1	Bag
2	Pail
3	Can
4	Big_Box
5	Small_Box
6	Fragile

Figure 21: Storage Location Assignment alternatives (Randomised, PLF based)



3.5.1.3 Applied routing algorithms

I needed to define PLF based routing algorithms(s) to examine how the system attributes influence the order picking lead time in operation. While the previous section (Section 3.4.) proved the effectiveness of my BMA algorithms for the OPRP-PLF, I chose effective algorithms from my algorithm alternatives.

While the StrictInit+NonStrictBM+BestDevLS-SA algorithm performed one of the most stable results, I applied this algorithm for evaluation as a non-strict algorithm. Furthermore, I would like to examine the necessity of the reconstruction in the case of different system attributes (e.g. structured SLA based on PLF). I applied a strict routing algorithm also to make me able to compare the order picking lead time on an objective way when UL reconstruction is allowed and forbidden. While the strict algorithms performed consistent results, I simply chose the strict pair of the non-strict algorithm, the StrictInit+StrictBM+StrictBestDevLS-SA.

I defined the previously described routing algorithm alternatives as follows:

Table 39: Algorithm alternatives - combinations of system attributes

Alternative	Algorithm	SLA	DA	Layout
NonStrict-PLF-left-big	Non-Strict	PLF	left	big
NonStrict-PLF-left-small	Non-Strict	PLF	left	small
NonStrict-PLF-middle-big	Non-Strict	PLF	middle	big
NonStrict-PLF-middle-small	Non-Strict	PLF	middle	small
NonStrict-PLF-right-big	Non-Strict	PLF	right	big
NonStrict-PLF-right-small	Non-Strict	PLF	right	small
NonStrict-RND-left-big	Non-Strict	RND	left	big
NonStrict-RND-left-small	Non-Strict	RND	left	small
NonStrict-RND-middle-big	Non-Strict	RND	middle	big
NonStrict-RND-middle-small	Non-Strict	RND	middle	small
NonStrict-RND-right-big	Non-Strict	RND	right	big
NonStrict-RND-right-small	Non-Strict	RND	right	small
Strict-PLF-left-big	Strict	PLF	left	big
Strict-PLF-left-small	Strict	PLF	left	small
Strict-PLF-middle-big	Strict	PLF	middle	big
Strict-PLF-middle-small	Strict	PLF	middle	small
Strict-PLF-right-big	Strict	PLF	right	big
Strict-PLF-right-small	Strict	PLF	right	small
Strict-RND-left-big	Strict	RND	left	big
Strict-RND-left-small	Strict	RND	left	small
Strict-RND-middle-big	Strict	RND	middle	big
Strict-RND-middle-small	Strict	RND	middle	small
Strict-RND-right-big	Strict	RND	right	big
Strict-RND-right-small	Strict	RND	right	small

- Non-Strict – StrictInit+NonStrictBM+BestDevLS-SA
- Strict – StrictInit+StrictBM+StrictBestDevLS-SA

3.5.2 Evaluated alternatives

The previous section defined several alternatives for the following system attributes:

- Layout dimension (2 alternatives)
- DA position (3 alternatives)
- SLA (2 alternatives)
- Routing algorithm (2 alternatives)

A combinations of these attributes defined the alternatives for evaluation. Table 39 summarises the 24 $(2 \cdot 3 \cdot 2 \cdot 2)$ alternatives. Examining every alternative is necessary to see the effects of every attributes in every circumstances.

3.5.3 Order picking lists

While the order picking list characteristics are important factors during the evaluation of an order picking system, I defined several different characteristics orders.

Generally almost every order contains every PC, only some simple and short lists contain one less than the every PC. The ordered quantity is simplified in the case study, most of the ordered quantity equals one layer of products, except some cases when low quantity (*canLQ*) POPC appears.

I aimed to define specific order characteristics which are industrially relevant and the examination is necessary. Based on this point of view, I defined order types ("Simple", "LQ", and "Border"). In the case of simple orders, the ordered products are allocated somewhere in the middle of the PLF based zones. The "LQ" orders contain low quantity orders (*canLQ*). The "Border" type of orders contain *canLQ* POPC and the ordered products are allocated near the borderline of the PLF based zones. Furthermore, I defined three different lengths of orders for each order type. I generated orders which contain 8, 10 or 12 ordered records [8].

I defined 9 ($3 \cdot 3$) order picking lists based on the previously described preferences as follows:

- Type of the order picking list - "Simple", "LQ", "Border"
- Length of the order picking list - 8, 10, 12 records

Figures 22, 23, and 25 demonstrate each order picking list in the case of Randomised SLA and PLF based SLA. The colour codes are the same as before in the case of the SLA visualisation (Table 38). The low quantity (*canLQ*) records are visualised as Fig. 24 shows. The figures highlight the structured and sequenced SLA on the right side. It is important to see, that the *canLQ* POPCs are positioned within the can zone, because the SLA is defined based on PC without order parameters and each product has only one picking position.

3.5.4 Simulation results and consequences

Each of the 24 alternatives have been evaluated based on each of the 9 order picking lists. Table 40 shows the calculated lead times for each alternative and list pairs, where the reconstruction times are marked in parentheses.

3.5.4.1 Necessity of the reconstruction

Table 41 compares the system attribute combinations from the routing point of view. The values are the summarised lead time of the 9 order picking lists, because evaluation is necessary based on a set of order. It generally highlights, that the Non-Strict routing algorithm can reach on average 5.37% lower lead time. However, it also highlights, that the Non-Strict algorithm was effective especially in the case of randomised SLA. The average difference of the routing alternatives in the case of PLF based SLA is 0.86%. It

Figure 22: Simple 8, Simple 10, and Simple 12 (from top to down) order picking lists are visualised in the case of Randomised SLA (left) and PLF based SLA (right)

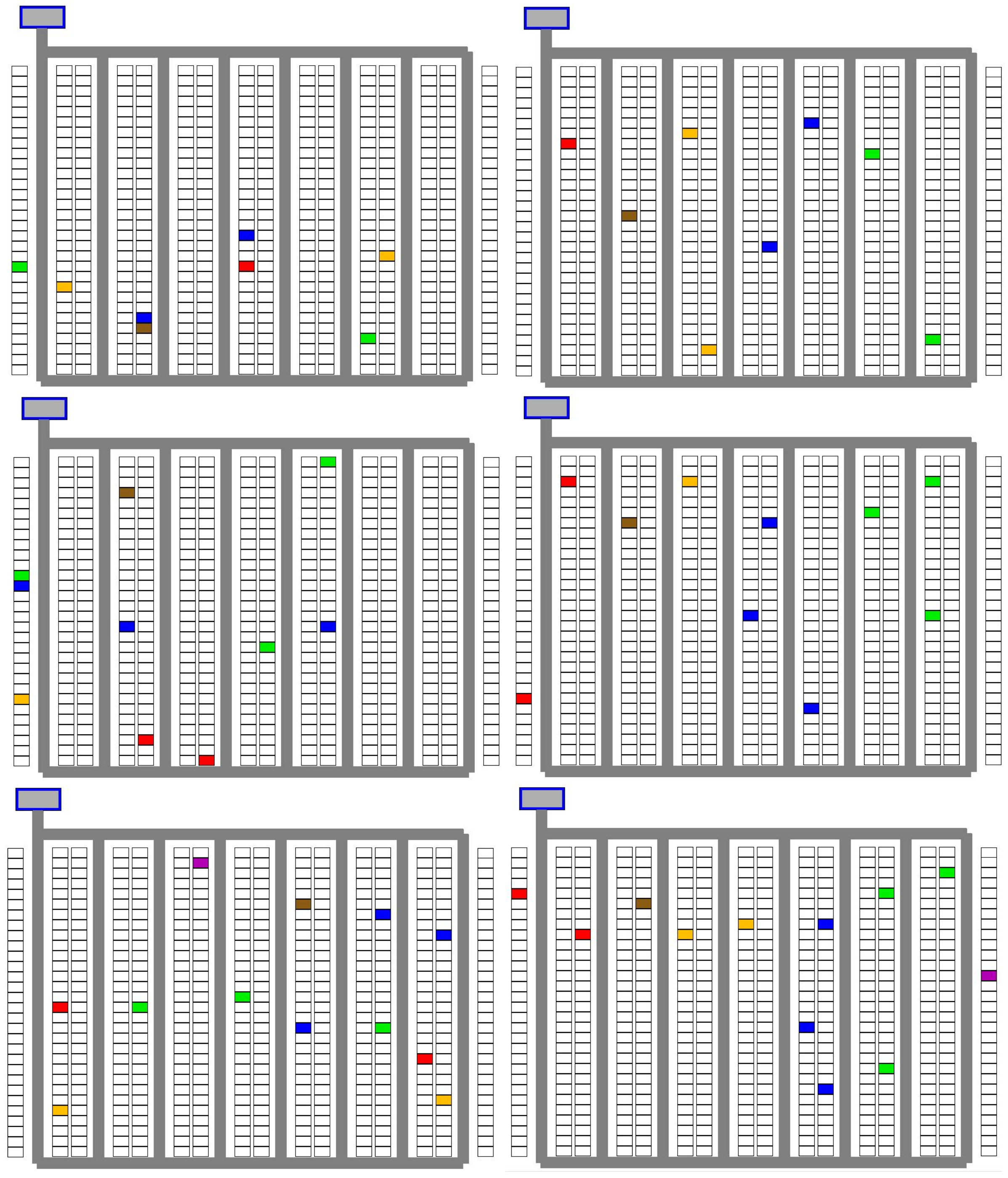


Figure 23: LQ 8, LQ 10, and LQ 12 (from top to down) order picking lists are visualised in the case of Randomised SLA (left) and PLF based SLA (right)

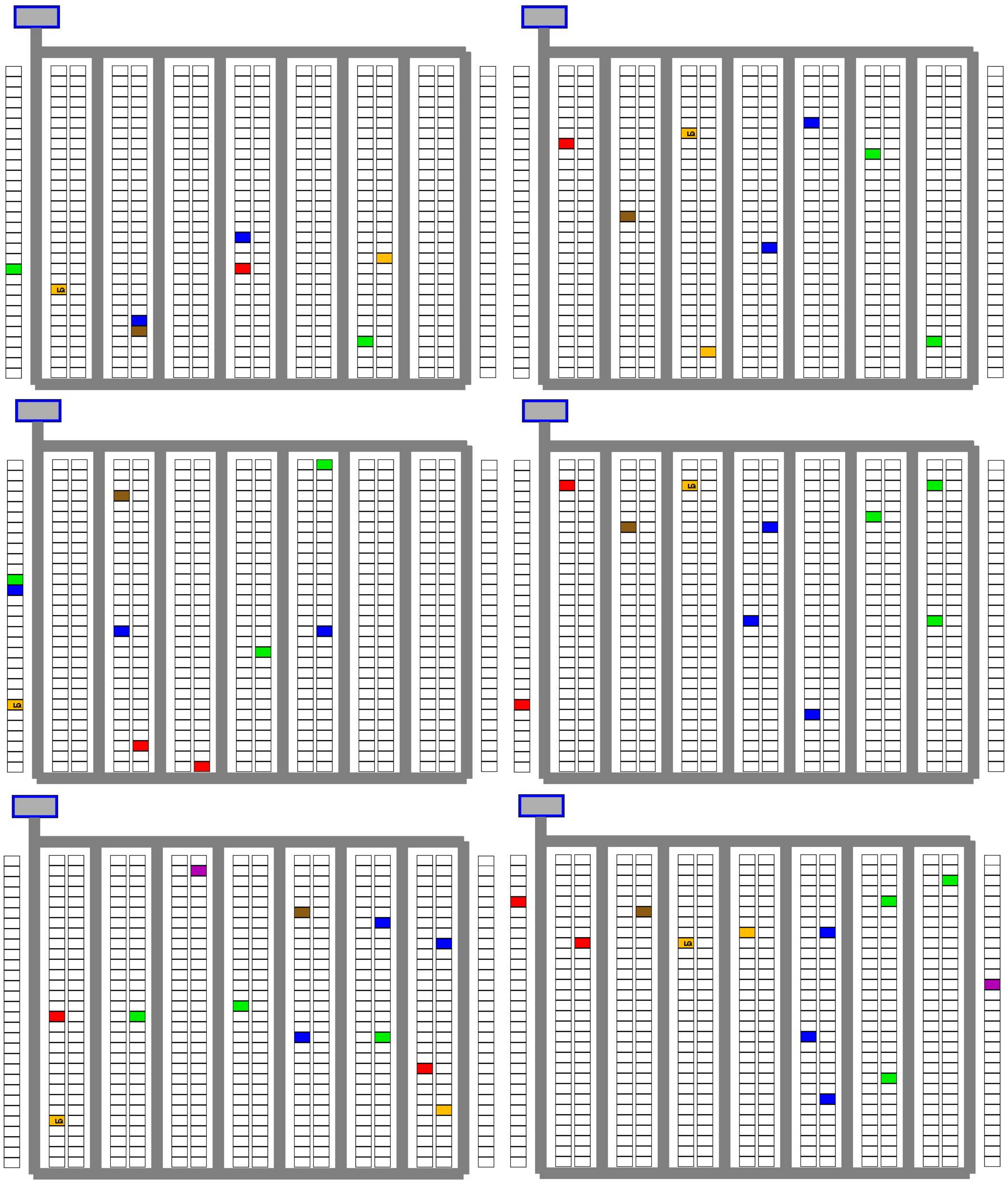


Figure 24: The visualisation of the Low Quantity (LQ) records

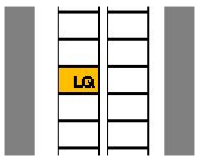


Figure 25: Border 8, Border 10, and Border 12 (from top to down) order picking lists are visualised in the case of Randomised SLA (left) and PLF based SLA (right)

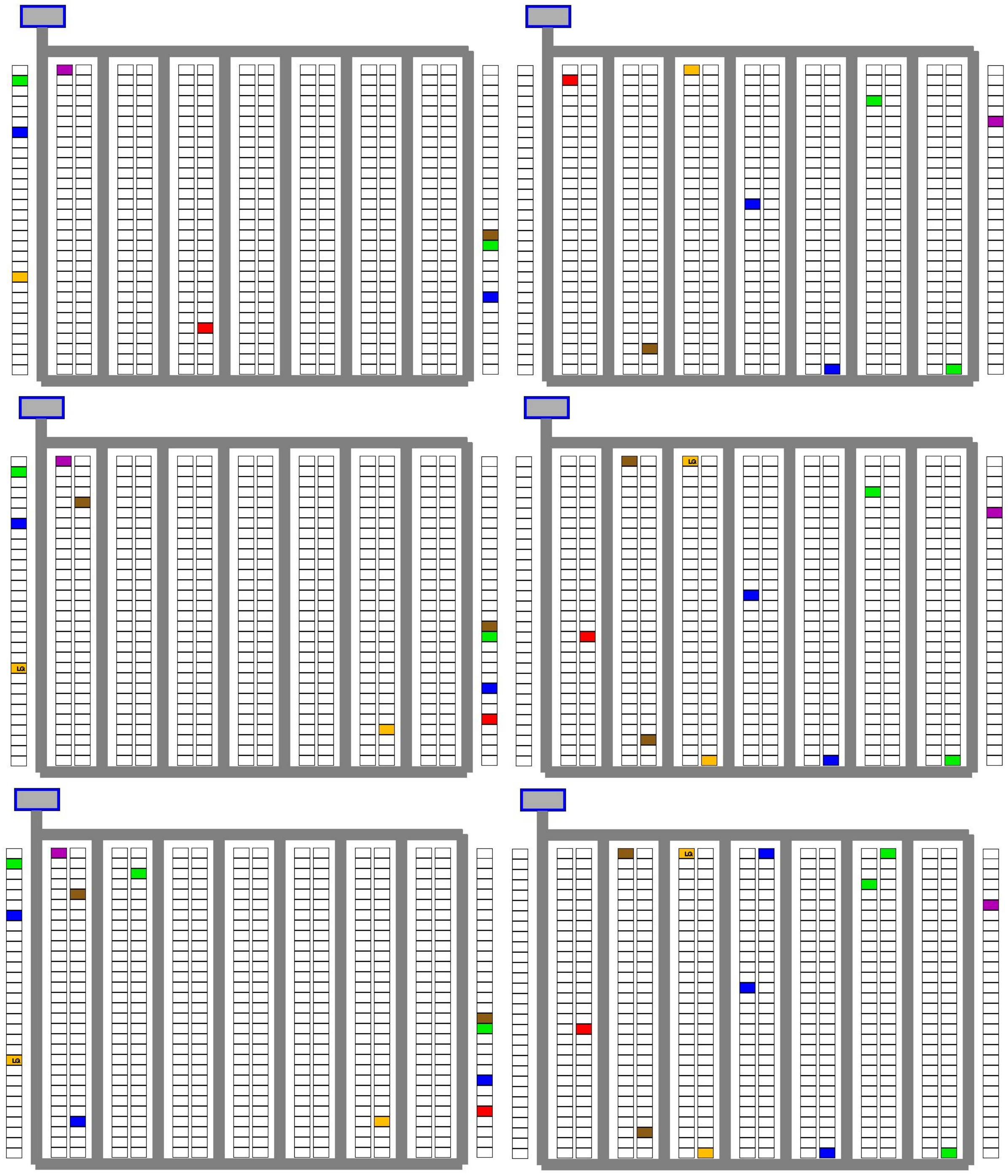


Table 40: Simulation results of the layout and SLA case study

Alternative	Simple 8	Simple 10	Simple 12	LQ 8	LQ 10	LQ 12	Border 8	Border 10	Border 12
NonStrict-PLF-left-big	5:45.5090 (0.0000)	6:14.3713 (0.0000)	7:00.8383 (0.0000)	5:58.2036 (0.0000)	6:39.7605 (0.0000)	8:00.8383 (1:00.0000)	5:27.2455 (30.0000)	6:37.3952 (1:15.0000)	7:25.1796 (15.0000)
NonStrict-PLF-left-small	3:32.7545 (0.0000)	3:57.1856 (0.0000)	4:30.4192 (0.0000)	3:39.1018 (0.0000)	4:09.8802 (0.0000)	5:00.6587 (0.0000)	3:35.4790 (15.0000)	4:17.5150 (15.0000)	4:50.0898 (15.0000)
NonStrict-PLF-middle-big	5:45.5090 (0.0000)	6:14.3713 (0.0000)	7:00.8383 (0.0000)	5:58.2036 (0.0000)	6:39.7605 (0.0000)	8:00.8383 (1:00.0000)	5:27.2455 (30.0000)	6:25.9581 (45.0000)	7:12.4850 (15.0000)
NonStrict-PLF-middle-small	3:32.7545 (0.0000)	3:57.1856 (0.0000)	4:30.4192 (0.0000)	3:39.1018 (0.0000)	4:09.8802 (0.0000)	5:00.6587 (0.0000)	3:35.4790 (15.0000)	4:11.1677 (15.0000)	4:43.7425 (15.0000)
NonStrict-PLF-right-big	6:09.2216 (0.0000)	6:27.0659 (0.0000)	7:00.8383 (0.0000)	6:17.3653 (0.0000)	6:52.4551 (0.0000)	8:00.8383 (1:00.0000)	5:44.9401 (15.0000)	6:25.9581 (45.0000)	7:12.4850 (15.0000)
NonStrict-PLF-right-small	3:44.6108 (0.0000)	4:03.5329 (0.0000)	4:30.4192 (0.0000)	3:48.6826 (0.0000)	4:16.2275 (0.0000)	5:00.6587 (0.0000)	3:36.8263 (0.0000)	4:11.1677 (15.0000)	4:43.7425 (15.0000)
NonStrict-RND-left-big	6:28.5629 (1:30.0000)	7:24.7605 (45.0000)	8:55.4192 (45.0000)	5:58.5629 (1:00.0000)	7:24.3713 (30.0000)	9:16.4970 (45.0000)	5:27.6347 (45.0000)	6:39.7904 (1:15.0000)	7:59.9401 (1:30.0000)
NonStrict-RND-left-small	4:09.2216 (0.0000)	4:43.7126 (0.0000)	5:50.2096 (45.0000)	4:01.4072 (15.0000)	4:44.0419 (15.0000)	5:53.7126 (30.0000)	3:46.3174 (45.0000)	4:35.7186 (45.0000)	5:32.9940 (30.0000)
NonStrict-RND-middle-big	6:26.4970 (1:15.0000)	6:52.3952 (30.0000)	8:54.2515 (1:00.0000)	5:56.4970 (45.0000)	6:48.4132 (1:15.0000)	9:23.9521 (1:00.0000)	5:41.6467 (45.0000)	7:12.0359 (1:30.0000)	8:28.2635 (2:00.0000)
NonStrict-RND-middle-small	4:10.4192 (0.0000)	4:31.1976 (30.0000)	5:51.2575 (30.0000)	3:47.6048 (0.0000)	4:31.3473 (15.0000)	6:00.0599 (30.0000)	3:53.3234 (45.0000)	4:54.7605 (45.0000)	5:45.4192 (45.0000)
NonStrict-RND-right-big	6:26.4970 (1:15.0000)	7:22.0060 (45.0000)	9:20.8084 (45.0000)	5:56.4970 (45.0000)	7:01.1078 (1:15.0000)	9:42.8144 (30.0000)	5:53.4132 (1:00.0000)	7:12.0359 (1:30.0000)	8:14.0719 (1:30.0000)
NonStrict-RND-right-small	4:23.1138 (0.0000)	4:46.3473 (30.0000)	5:57.6048 (30.0000)	4:00.2994 (0.0000)	4:44.0419 (15.0000)	6:06.4072 (30.0000)	4:06.7066 (1:00.0000)	4:55.5389 (45.0000)	5:45.4192 (45.0000)
Strict-PLF-left-big	5:45.5090 (0.0000)	6:14.3713 (0.0000)	7:00.8383 (0.0000)	5:58.2036 (0.0000)	6:39.7605 (0.0000)	8:01.3174 (0.0000)	5:53.6527 (0.0000)	6:57.3653 (0.0000)	7:42.5150 (0.0000)
Strict-PLF-left-small	3:32.7545 (0.0000)	3:57.1856 (0.0000)	4:30.4192 (0.0000)	3:39.1018 (0.0000)	4:09.8802 (0.0000)	5:00.6587 (0.0000)	3:36.8263 (0.0000)	4:18.6826 (0.0000)	4:51.2575 (0.0000)
Strict-PLF-middle-big	5:45.5090 (0.0000)	6:14.3713 (0.0000)	7:00.8383 (0.0000)	5:58.2036 (0.0000)	6:39.7605 (0.0000)	8:01.3174 (0.0000)	5:53.6527 (0.0000)	6:44.6707 (0.0000)	7:29.8204 (0.0000)
Strict-PLF-middle-small	3:32.7545 (0.0000)	3:57.1856 (0.0000)	4:30.4192 (0.0000)	3:39.1018 (0.0000)	4:09.8802 (0.0000)	5:00.6587 (0.0000)	3:36.8263 (0.0000)	4:12.3353 (0.0000)	4:44.9102 (0.0000)
Strict-PLF-right-big	6:09.2216 (0.0000)	6:27.0659 (0.0000)	7:00.8383 (0.0000)	6:17.3653 (0.0000)	6:52.4551 (0.0000)	8:01.3174 (0.0000)	5:53.6527 (0.0000)	6:44.6707 (0.0000)	7:29.8204 (0.0000)
Strict-PLF-right-small	3:44.6108 (0.0000)	4:03.5329 (0.0000)	4:30.4192 (0.0000)	3:48.6826 (0.0000)	4:16.2275 (0.0000)	5:00.6587 (0.0000)	3:36.8263 (0.0000)	4:12.3353 (0.0000)	4:44.9102 (0.0000)
Strict-RND-left-big	6:58.4431 (0.0000)	7:47.4251 (0.0000)	10:17.8443 (0.0000)	6:53.2934 (0.0000)	8:05.0299 (0.0000)	10:10.2994 (0.0000)	6:57.6048 (0.0000)	9:27.1856 (0.0000)	10:06.1078 (0.0000)
Strict-RND-left-small	4:09.2216 (0.0000)	4:43.7126 (0.0000)	6:08.9222 (0.0000)	4:06.6467 (0.0000)	4:52.5150 (0.0000)	6:05.1497 (0.0000)	4:08.8024 (0.0000)	5:33.5928 (0.0000)	6:03.0539 (0.0000)
Strict-RND-middle-big	7:00.8383 (0.0000)	8:00.7186 (0.0000)	10:30.5389 (0.0000)	6:15.2096 (0.0000)	7:31.2575 (0.0000)	10:22.9940 (0.0000)	7:10.2994 (0.0000)	9:27.1856 (0.0000)	10:06.1078 (0.0000)
Strict-RND-middle-small	4:10.4192 (0.0000)	4:50.3593 (0.0000)	6:15.2695 (0.0000)	3:47.6048 (0.0000)	4:35.6287 (0.0000)	6:11.4970 (0.0000)	4:15.1497 (0.0000)	5:33.5928 (0.0000)	6:03.0539 (0.0000)
Strict-RND-right-big	7:26.2275 (0.0000)	8:26.1078 (0.0000)	10:43.2335 (0.0000)	6:40.5988 (0.0000)	7:56.6467 (0.0000)	10:35.6886 (0.0000)	7:48.3832 (0.0000)	9:27.1856 (0.0000)	10:06.1078 (0.0000)
Strict-RND-right-small	4:23.1138 (0.0000)	5:03.0539 (0.0000)	6:21.6168 (0.0000)	4:00.2994 (0.0000)	4:48.3234 (0.0000)	6:17.8443 (0.0000)	4:34.1916 (0.0000)	5:33.5928 (0.0000)	6:03.0539 (0.0000)

is 9.89% in the case of Randomised SLA. It highlights, that reconstruction is necessary in both cases, but it is more important in the case of randomised SLA.

Table 41: Comparing the summarised order picking times of Non-Strict and Strict solutions

Parameter combination	Non-Strict	Strict	Strict - Non-Strict difference	Strict - Non-Strict difference %
PLF-left-big	0:59:05	1:00:09	0:01:04	1.77%
PLF-left-small	0:37:29	0:37:32	0:00:03	0.13%
PLF-middle-big	0:58:40	0:59:43	0:01:03	1.76%
PLF-middle-small	0:37:16	0:37:19	0:00:03	0.13%
PLF-right-big	1:00:06	1:00:52	0:00:46	1.26%
PLF-right-small	0:37:51	0:37:53	0:00:02	0.09%
RND-left-big	1:05:30	1:16:40	0:11:10	14.57%
RND-left-small	0:43:13	0:45:47	0:02:34	5.61%
RND-middle-big	1:05:40	1:16:21	0:10:41	13.99%
RND-middle-small	0:43:22	0:45:39	0:02:17	5.00%
RND-right-big	1:07:06	1:19:07	0:12:01	15.19%
RND-right-small	0:44:42	0:47:02	0:02:20	4.96%
AVG	**0:51:40**	**0:55:20**	**0:03:40**	**5.37%**

Table 42 highlights the differences between the Non-Strict and Strict algorithms order by order. The values show how much higher the lead time of the strict alternatives is. It highlights, that the longer and more complex lists make the reconstruction necessary in the case of PLF based SLA. When the differences are examined order by order, the Non-Strict algorithm can reach on average 2.33% savings on the lead time in the case of PLF based SLA. The maximum difference is 7.4%. Furthermore, the reconstruction results in high, on average 10.84% savings in the case of randomised SLA. The maximum saving is 29.63%. It also emphasises, that reconstruction is more effective on randomised SLA, but it can considerably decrease the lead time in the case of PLF based SLA, too.

Table 42: Highlighting the differences between the Non-Strict and Strict algorithms order by order.

ScenarioOrigin	Simple 8	Simple 10	Simple 12	LQ 8	LQ 10	LQ 12	Border 8	Border 10	Border 12
Strict-PLF-left-big						0.21%	7.37%	4.80%	3.68%
Strict-PLF-left-small							0.46%	0.39%	0.34%
Strict-PLF-middle-big						0.21%	7.37%	4.70%	3.79%
Strict-PLF-middle-small							0.46%	0.40%	0.35%
Strict-PLF-right-big						0.21%	2.55%	4.70%	3.79%
Strict-PLF-right-small								0.40%	0.35%
Strict-RND-left-big	7.18%	4.93%	13.29%	13.32%	8.45%	8.85%	21.58%	29.63%	20.96%
Strict-RND-left-small			4.89%	2.03%	2.74%	3.29%	8.87%	17.42%	8.54%
Strict-RND-middle-big	8.10%	14.17%	15.24%	5.07%	9.53%	9.49%	20.70%	23.81%	16.17%
Strict-RND-middle-small		6.55%	6.40%		1.45%	2.96%	8.63%	11.71%	4.96%
Strict-RND-right-big	13.45%	12.65%	12.91%	11.00%	11.55%	8.35%	24.57%	23.81%	18.48%
Strict-RND-right-small		5.61%	6.30%		1.39%	2.92%	10.22%	11.41%	4.96%

Obviously the results (Table 40) highlight the “Small” layout requires less reconstruction because of the shorter distances. However, more complex (e.g. “Border”) lists reached the lowest lead time with reconstruction in the case of “Small” layout, too.

The average reconstruction time was 44 seconds in the case of randomised SLA and 11 seconds in the case of PLF based SLA. It also highlights, that reconstruction is more important in the case of randomised SLA, but it is also necessary when the SLA is structured based on PLF.

3.5.4.2 Necessity of the PLF based SLA

While the results highlighted the effectiveness of the Non-Strict alternatives, I examined the effects of the SLA alternatives on the Non-Strict alternatives. The average of the summarised lead time of PLF based Non-strict alternatives is 0:48:25. Furthermore, it is 0:54:56 for the randomised alternatives. It shows 11.86% difference. The same examination shows 20.81% in the case of strict alternatives. I can conclude, that the PLF based SLA has a significant impact on the order picking lead time.

3.5.4.3 Effects of the DA location positioning on lead time

I will describe the effects of the DA position on the picking sequence and the order picking lead time based on order picking the "Border 10" list on "Big" layout, where the PLF based SLA is applied. I concentrate on the "Big" layout because it highlights a more significant effect because of the higher distances, but the statements are relevant on the "Small" layout, too. Table 43 summarises the simulation results of the highlighted cases.

Figures 26 and 27 highlight the picking sequence of "Border 10" order with strict routing. The strict routing results in the same sequence in both left and right DA position. The difference between the order picking lead time of Strict-PLF-right-big and Strict-PLF-left-big caused by the different travelled distance because of the DA positions.

Otherwise the Non-Strict routing algorithm resulted in a different picking sequence depending on the DA position (Figures 28 and 29). It caused different reconstruction and travelling time. NonStrict-PLF-right-big required the reconstruction of three ordered products on two positions. The first reconstruction was made after picking the first "Pail" record (brown position) and before picking the "Bag" record (red position). The second reconstruction was made after picking the "CanLQ" (orange position) and "SmallBox" (green position) records and before the second "BigBox" record. The reconstruction time happened twice here, because the "BigBox" should be positioned on the UL under both the "CanLQ" and "SmallBox" records. NonStrict-PLF-left-big required five reconstructions on three positions. The first reconstruction was the same as in the previous case. The second reconstruction was made after the "Fragile" record (purple position) and before the second "SmallBox" record. Furthermore, the third reconstruction took three reconstruction times because the "CanLQ" record should be positioned on the UL under the "Fragile" and the two "SmallBox" records. Obviously the different picking sequence resulted in a different travel time, too.

However, NonStrict-PLF-left-big resulted in a lower travel time. The higher reconstruction time caused higher lead time than NonStrict-PLF-right-big in the case of the

"Border 10" list. It highlights, that it is not enough to minimise the routing distance. It is important to minimise the picking lead time, especially when reconstruction is allowed. The results of DA position examination highlight, that the positioning of the DA position has a significant impact on the order picking lead time even with Strict and Non-Strict routing.

Table 43: Effects of the DA position on the order picking lead time of "Border 10" list

Border 10	Lead time	Picking time	Reconstruction time	Travel time
Strict-PLF-right-big	6:44.67	1:40.00	0.00	5:04.67
Strict-PLF-left-big	6:57.36	1:40.00	0.00	5:17.36
NonStrict-PLF-right-big	6:25.95	1:40.00	0:45.00	4:00.95
NonStrict-PLF-left-big	6:37.39	1:40.00	1:15.00	3:42.39

Figure 26: Picking sequence of the "Border 10" list in the case of Strict-PLF-right-big combination

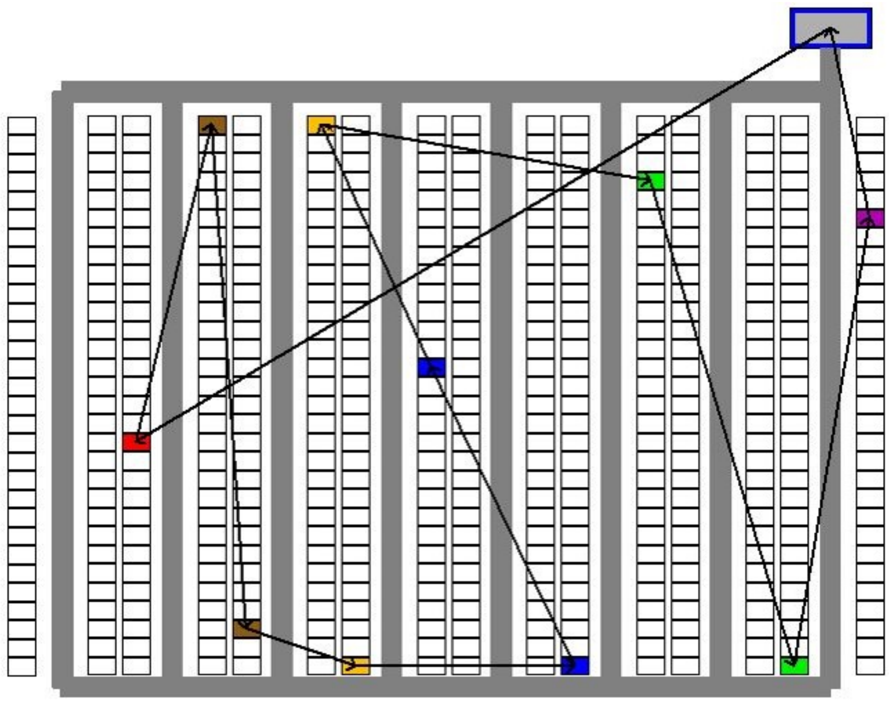


Figure 27: Picking sequence of the “Border 10” list in the case of Strict-PLF-left-big combination

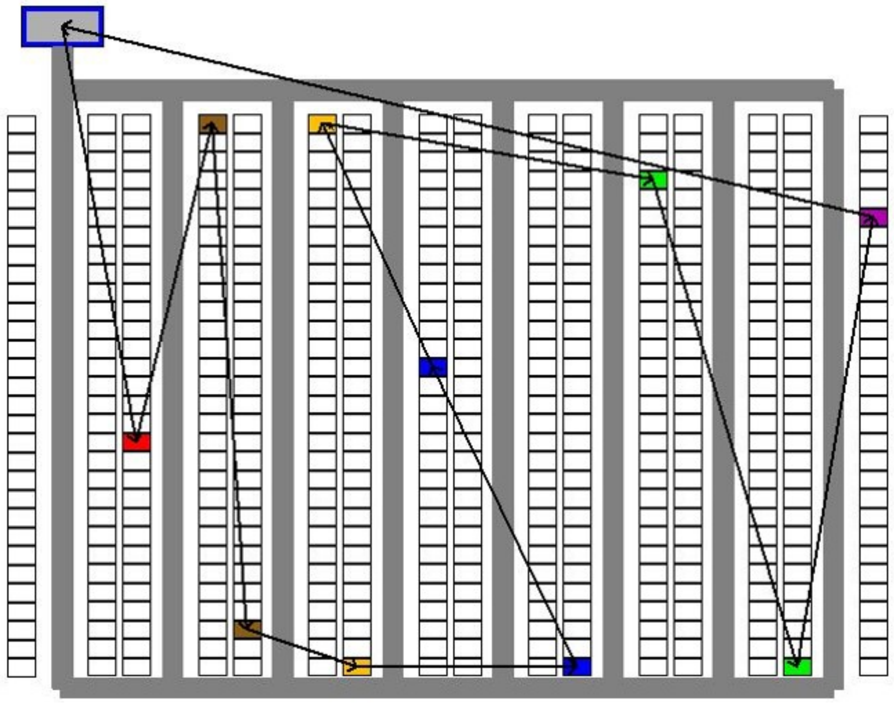


Figure 28: Picking sequence of the “Border 10” list in the case of NonStrict-PLF-right-big combination

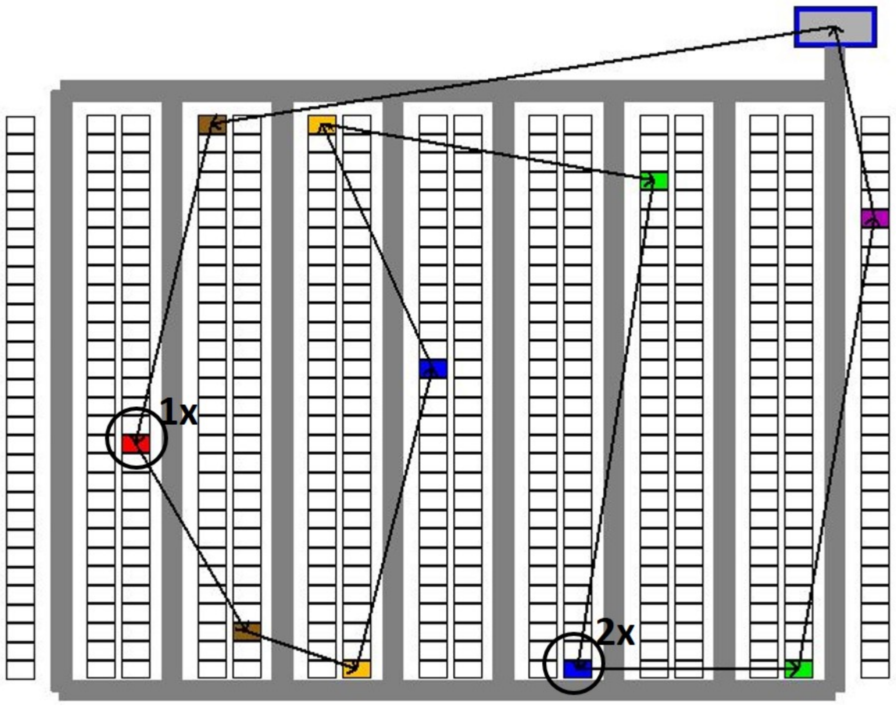
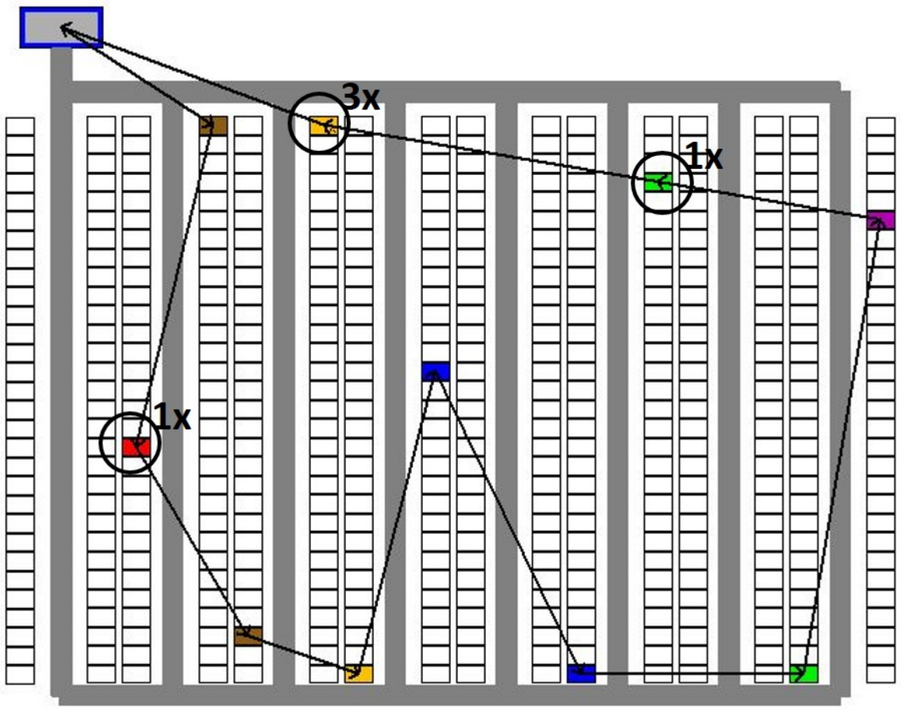


Figure 29: Picking sequence of the "Border 10" list in the case of NonStrict-PLF-left-big combination



The results show, that the lowest lead times are reached in my case study in the case of Non-Strict routing and PLF based SLA. I can conclude, that the SLA based on PLF can significantly decrease the order picking lead time when PLF is a relevant factor of the warehouse. I realised, that reconstruction could decrease the order picking lead time even in the case of PLF based SLA because of the order characteristics. It could be necessary in particular industrial cases, like ordering products from the borderline of the PLF zones or ordering low quantity ($CanLQ$). My case study also highlighted, that the layout dimensions could influence the necessary order picking algorithm and the DA position has a significant impact on the lead time independently from the routing algorithm and SLA.

3.5.5 Summarising statements

While several factors influence the warehouse operations, it is necessary to harmonise those factors for the purpose of effective warehousing processes. Relying upon this fact, I examined the effect of the warehouse layout attributes, the Storage Location Assignment (SLA) and the unit load reconstruction on the order picking lead time.

Thesis statement 5. *I proved, that when Pallet Loading Feature is relevant, then Pallet Loading Feature based SLA results in a lower lead time. My simulation results also proved, that allowing reconstruction is necessary for order picking lead time minimisation even in the case of Pallet Loading Feature based SLA because of the order characteristics and the location of the Departure and Arrival position. I verified, that the Departure and Arrival position has an impact on the travelling and reconstruction times in the case of OPRP-PLF, unlike classical routing problems (e.g. TSP).*

My publication related to the statement: [8].

Chapter 4

Summary

I realised during my industrial activities and state of the art research, that there is a lack of implementing unit load building aspects into order picking routing and storage location assignment optimisation. The unit load building rules, the allowed or forbidden picking sequences, the applied unit load reconstruction during order picking, and the unit load building rule based layout design and storage location assignment could have a significant impact on the order picking effectiveness.

My state of the art research (Section 2.1) highlighted, that many solutions have been defined for harmonising SLA and routing to decrease the routing distances and times, but the physical product parameters (dimensions, weight, packaging), the product stacking attributes, and the order characteristics are not considered comprehensively in order to build stable ULs based on physically possible order picking sequence. From another perspective, many researchers have attained valuable results in the fields of Pallet Loading and Bin Packing Problems, but the solutions are rarely harmonised with SLA and order picking routing algorithms.

My general goal was to find industrially relevant and novel scientific methodologies and algorithms to support pickers to minimise the order picking lead time, to build stable transport units, and to avoid product damages when product stacking aspects are relevant at the given warehouse.

I highlighted and defined a novel and complex sub-problem of the Order Picking Routing Problem and proved its necessity. I called it Order Picking Routing Problem based on Pallet Loading Feature (OPRP-PLF). I defined the Pallet Loading Features (PLF) as a logistics system attribute. I collected the influencing factors and aspects (product attributes, order picking list characteristics, and order picking system) of the PLF. I built a formalised, flexible, parametric, and industrially relevant model for the OPRP-PLF based on known, easily measurable, and rarely changing data. I formalised the logical picking possibilities of the order picking records by Pallet Loading Feature based Decision Matrix. I developed methodologies for examining, when it is necessary to implement an OPRP-PLF algorithm at a warehouse. I defined the Pallet Loading Rate and highlighted the necessity of the order picking process monitoring.

I made complexity evaluation for the industrially relevant cases of the OPRP-PLF and highlighted those exponential complexity. I proved, that when meta-heuristics optimisation methodology is necessary for OPRP-PLF to find an acceptable picking sequence within the available running time.

I formalised an objective function for the OPRP-PLF to evaluate order picking sequence variation. It considers the unit load reconstruction times besides travelling and picking times. I highlighted based on analytic examination of simple cases, that considering reconstruction during OPRP-PLF is necessary to minimise the order picking lead time, because the shortest route could cause more reconstruction and higher lead time. I proposed, that supporting the pickers by algorithms is necessary in the case of complex OPRP-PLF.

I applied Bacterial Memetic Algorithm (BMA) and Simulated Annealing (SA) algorithms as novel approaches for the OPRP-PLF. I defined several alternatives for the BMA operators for comparison. I improved BMA operators (bacterial mutation and local search) by simulated annealing methodology. Using the BMA operators with SA methodology is a novelty of the proposed algorithms, whose effectiveness has been evaluated.

I defined possible algorithms based on combination of my BMA operators and evaluated those with the SA algorithms by 10, 20, and 50 record long order picking lists. The simulation results highlighted, that allowance of reconstruction could result in a lower order picking lead time, but the initial population should be generated without reconstruction (strict sequence). I proved, that applying BMA for OPRP-PLF is more effective than the SA algorithms. Furthermore, I highlighted the operability of SA combined BMA operators in the case of less than 20 record long lists. Finally, I defined a right combination of the BMA operators and the parameter values of BMA algorithm for further applications.

I made 24 industrially relevant order picking system alternatives based on layout dimensions, positioning of the departure and arrival position, storage location assignment and allowance of unit load reconstruction. I evaluated the alternatives by order picking lists with different nature and length. My simulation results of the industrial cases proved, that the PLF based SLA, the allowed unit load reconstruction, and the departure and arrival position have a significant impact on the order picking lead time when PLF is relevant at the warehouse.

My research highlighted and formalised the OPRP-PLF, and developed effective OPRP-PLF algorithms. The developed solutions could provide significant results in supporting the pickers by defining realistic picking sequence within the possible time window. The proposed solutions are important, where the variability of packages, the ordered quantities, and the order picking system itself make the stable unit load building a combinatorially complex problem. The defined BMA algorithm could be integrated into any warehouse management systems (WMS) as a routing algorithm. It can also work as a connected external optimisation module triggered by an order picking list via any interface and it sends back the optimised list for the WMS. Besides the advantages and effectiveness of the BMA solution, its sophisticated structure and complex parameter setting requires BMA experienced experts for implementation.

As further research I would like to implement the Special Product and Order Parameter Class (SPOPC) issues into the algorithms to model the system characteristics in a more realistic way. The SPOPC considers the previously picked units and their sequence to define the behaviour of the next record (Section 3.1.2.3). While my proposed research highlighted the importance of PLF based SLA, it applied previously defined SLA based on logical, manual, and static methodology. I would like to develop SLA algorithms, which will be able to update the SLA based on the actual order characteristics and the position occupation. It would be useful during order picking stock replenishment and storing in processes to find the actually relevant picking or storage position for the products. Although I proved, that applying unit load reconstruction could decrease the order picking lead time, it should be limited to minimise the product damage possibilities in the case of sensitive products. I might apply some fuzzy methodology into the reconstruction procedure. While this work discussed PLF based order picking routing optimisation of one UL, complementing the algorithm by separating the purchased order for ULs based on PLF would be a more complex and industrially important problem. The aim of the extended algorithm complemented by order separation should be the minimisation of the lead time of the whole order performing process.

Bibliography

[1] Design Criteria for Ammunition Unit Loads - MIL-STD-1660. Technical report, U.S. DOD, Dept of the Navy, Naval Sea Systems Command, Washington, 1970.

[2] R. Alvarez-Valdes, F. Parreño, and J. M. Tamarit. A tabu search algorithm for the pallet loading problem. *OR Spectrum*, 27:43–61, 2005.

[3] K. Balázs. *Advanced Approaches in the Application Methodologies of Evolutionary Algorithms*. PhD thesis, Budapest University of Technology and Economics, 2013.

[4] S. Bangsow. *Tecnomatix Plant Simulation - Modeling and Programming by Means of Examples*. Springer, 2015.

[5] E. E. Bischoff. Three-dimensional packing of items with limited load bearing strength. *European Journal of Operational Research*, 168(3):952–966, 2006.

[6] T. Bódis and J. Botzheim. Modelling order picking sequencing variations of pallet setup clusters. In *Proc. of The International Conference on Logistics and Sustainable Transport 2015*, pages 86–94, Celje, Slovenia, June 2015.

[7] T. Bódis and J. Botzheim. A simple case of pallet setup features based order picking routing optimization. *Acta Technica Jaurinensis*, 9(3):204–215, 2016.

[8] T. Bódis and J. Botzheim. Stacking property based storage location assignment for minimising order picking lead time. In *3rd Interdisciplinary Conference on Production, Logistics and Traffic*, pages 1–6, Darmstadt, Germany, 2017.

[9] T. Bódis and J. Botzheim. Bacterial Memetic Algorithms For Order Picking Routing Problem With Loading Constraints. *Expert Systems with Applications*, 105:196–220, 2018.

[10] T. Bódis, K. Udvardy, and J. Botzheim. Interactive training and modeling environment for considering pallet setup features in storage location assignment of order picking zone. In *Proceedings of the Mecatronics-2014-Tokyo*, pages 64–69, 2014.

[11] T. Bódis, J. Botzheim, and P. Földesi. Necessity and Complexity of Order Picking Routing Optimisation based on Pallet Loading Features. *Acta Univ. Sapientiae, Informatica*, 9(2):162–194, 2017.

[12] M. Bóna. *A Walk Through Combinatorics: An Introduction to Enumeration and Graph Theory - Third Edition*. World Scientific Publishing Co. Pte. Ltd., Singapore, 2011. ISBN 978-981-4335-23-2.

[13] A. Bortfeldt and G. Wäscher. Constraints in container loading – a state-of-the-art review. *European Journal of Operational Research*, 229(1):1–20, 2013.

[14] J. Botzheim. *Intelligens számítástechnikai modellek identifikációja evolúciós és gradiens alapú tanuló algoritmusokkal (in Hungarian)*. PhD thesis, Budapest University of Technology and Economics, 2007.

[15] J. Botzheim, M. Drobics, and L. T Kóczy. Feature selection using bacterial optimization. In *Proceedings of the international conference on information processing and management of uncertainty in knowledge-based systems, IPMU*, pages 797–804, 2004.

[16] J. Botzheim, C. Cabrita, L. T Kóczy, and A. E. Ruano. Fuzzy rule extraction by bacterial memetic algorithms. *International Journal of Intelligent Systems*, 24(3): 312–339, 2009.

[17] J. Botzheim, Y. Toda, and N. Kubota. Bacterial memetic algorithm for simultaneous optimization of path planning and flow shop scheduling problems. *Artificial Life and Robotics*, 17(1):107–112, 2012.

[18] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99: 300–313, 2016.

[19] F. T. Chan and H. Chan. Improving the productivity of order picking of a manual-pick and multi-level rack distribution warehouse through the implementation of class-based storage. *Expert Systems with Applications*, 38:2686–2700, 2011.

[20] T. L. Chen, C. Y. Cheng, Y. Y. Chen, and L. K. Chan. An efficient hybrid algorithm for integrated order batching, sequencing and routing problem. *International Journal of Production Economics*, 159:158–167, 2015.

[21] K. L. Choy, H. Y. Lam, L. Canhong, and C. K. H. Lee. A hybrid decision support system for storage location assignment in the fast-fashion industry. In *Technology Management in the IT-Driven Services (PICMET), 2013 Proceedings of PICMET '13*, pages 468–473, San Jose, California, USA, August 2013.

[22] Y. F. Chuang, H. T. Lee, and S. W. Tan. A robust heuristic method on the clustering-assignment problem model. *Computers & Industrial Engineering*, 98:63–67, 2016.

[23] Y. F. F. Chuang, H. T. T. Lee, and Y. C. C. Lai. Item-associated cluster assignment model on storage allocation problems. *Computers and Industrial Engineering*, 63(4): 1171–1177, 2012.

[24] Á. Csík and J. Botzheim. Multi-objective optimization of building envelopes by bacterial memetic algorithms. In *2013 World Congress on Nature and Biologically Inspired Computing (NaBIC)*, pages 245–252, August 2013.

[25] Á. Csík, J. Botzheim, J. Balázs, T. Csoknyai, and J. Hontvári. Energy and cost optimal design for the reconstruction of residential building envelopes by bacterial memetic algorithms. In *Proceedings of the 6th International Conference on Soft Computing and Intelligent Systems and the 13th International Symposium on Advanced Intelligent Systems, SCIS & ISIS 2012*, pages 1226–1231, Kobe, Japan, November 2012.

[26] Y. P. Cui, C. Yaodong, and T. Tianbing. Sequential heuristic for the two-dimensional bin-packing problem. *European Journal of Operational Research*, 240(1):43–53, 2015.

[27] D. Dasgupta. *Artificial Immune Systems and Their Applications.* Springer-Verlag, 1999.

[28] A. S. Dijkstra and K. J. Roodbergen. Exact route-length formulas and a storage location assignment heuristic for picker-to-parts warehouses. *Transportation Research Part E: Logistics and Transportation Review*, 102:38–59, 2017.

[29] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 26(1):29–41, 1996.

[30] R. Eberhart, Y. Shi, and J. Kennedy. *Swarm Intelligence.* Morgan Kaufmann, 2001.

[31] S. Edelkamp and S. Schrödl. *Heuristic search: theory and applications.* Morgan Kaufmann, 2012.

[32] P. Földesi, J. Botzheim, and L. T. Kóczy. Eugenic bacterial memetic algorithm for fuzzy road transport traveling salesman problem. *International Journal of Innovative Computing, Information and Control*, 7(5b):2775–2798, 2011.

[33] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, USA, 1989.

[34] K. Helsgaun. An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, 2000.

[35] J. H. Holland. *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence.* MIT Press, 1992.

[36] C.H. Hsu, C.S. Tsou, and F.J. Yu. Multicriteria tradeoffs in inventory control using memetic particle swarm optimization. *International Journal of Innovative Computing, Information and Control*, 5(11a):3755–3768, 2009.

[37] Z. C. Johanyak. Clonal selection based parameter optimization for sparse fuzzy systems. In *Proceedings of the IEEE 16th International Conference on Intelligent Engineering Systems*, pages 369–373, Lisbon, Portugal, 2012.

[38] D. Karaboga and B. Basturk. A powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm. *Journal of Global Optimization*, 39(3):459–471, 2007.

[39] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of the 1995 IEEE International Conference on Neural Networks*, pages 1942–1948, Perth, Australia, 1995.

[40] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[41] R. De Koster. How to assess a warehouse operation in a single tour. Technical report, RSM Erasmus University, Rotterdam, Netherlands, 2004.

[42] R. De Koster, T. Le-Duc, and K. J. Roodbergen. Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, 182: 481–501, 2007.

[43] H. C. W. Lau, T. M. Chan, W. T. Tsui, G. T. S. Ho, and K. L. Choy. An AI approach for optimizing multi-pallet loading operations. *Expert Systems with Applications*, 36 (3):4296–4312, 2009.

[44] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. *The Traveling salesman problem : a guided tour of combinatorial optimization.* Wiley, Chichester, 1985. ISBN 0471904139.

[45] R. A. Mahale and S. D. Chavan. A Survey: Evolutionary and Swarm Based Bio-Inspired Optimization Algorithms. *International Journal of Scientific and Research Publications*, 2(12), 2012.

[46] G. H. A. Martins and R. F. Dell. Solving the pallet loading problem. *European Journal of Operational Research*, 184(2):429–440, 2008.

[47] M. Matusiak, R. De Koster, L. Kroon, and J. Saarinen. A fast simulated annealing method for batching precedence-constrained customer orders in a warehouse. *European Journal of Operational Research*, 236(3):968–977, 2014.

[48] K. Moeller. Increasing warehouse order picking performance by sequence optimization. *Procedia Social and Behavioral Sciences*, 20:177–185, 2011.

[49] B. Molnár and Gy. Lipovszki. Multi-objective routing and scheduling of order pickers in a warehouse. *International Journal of Simulation: Systems, Science and Technology*, 6(5):22–33, 2005.

[50] P. Moscato. *On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms.* 1989.

[51] N. E. Nawa and T. Furuhashi. Fuzzy system parameters discovery by bacterial evolutionary algorithm. *IEEE Transactions on Fuzzy Systems*, 7(5):608–616, 1999.

[52] T. Öncan. MILP formulations and an Iterated Local Search Algorithm with Tabu Thresholding for the Order Batching Problem. *European Journal of Operational Research*, 243(1):142–155, 2015.

[53] J. C. H. Pan, P. H. Shih, and M. H. Wu. Storage assignment problem with travel distance and blocking considerations for a picker-to-part order picking system. *Computers & Industrial Engineering*, 62(2):527–535, 2012.

[54] J. C. H. Pan, P. H. Shih, and M. H. Wu. Order batching in a pick-and-pass warehousing system with group genetic algorithm. *Omega*, 57:238–248, 2015.

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

[56] H. Pollaris, K. Braekers, A. Caris, G. K. Janssens, and S. Limbourg. Vehicle routing problems with loading constraints: state-of-the-art and future directions. *OR Spectrum*, 37(2):297–330, 2014.

[57] K. J. Roodbergen. *Layout and routing methods for warehouses.* PhD thesis, RSM Erasmus University, the Netherlands, 2001.

[58] A Rushton, P. Croucher, and P. Baker. *The Handbook of Logistics and Distribution Management: Understanding the Supply Chain.* Kogan Page Limited, Great Britain, London, 2011. ISBN 0749457147.

[59] A. A. Saputra, J. Botzheim, and N. Kubota. Walking speed control in human behavior inspired gait generation system for biped robot. In *Proceedings of the 2016 IEEE Congress on Evolutionary Computation*, pages 4895–4902, Vancouver, Canada, July 2016.

[60] R. Saraiva. A layer-building algorithm for the three-dimensional multiple bin packing problem: a case study in an automotive company. *IFAC-PapersOnLine*, 48(3):490–495, 2015.

[61] W. J. Sawaya and W. C. Giauque. *Production and operations management.* Harcourt Brace Jovanovich, 1986. ISBN 0155719688.

[62] A. Scholz, S. Henn, M. Stuhlmann, and G. Wäscher. A new mathematical programming formulation for the Single-Picker Routing Problem. *European Journal of Operational Research*, 253(1):68–84, 2016.

[63] J. Y. Shiau and M. C. Lee. A warehouse management system with sequential picking for multi-container deliveries. *Computers and Industrial Engineering*, 58:382–392, 2010.

[64] J. Y. Shiau and H. L. Ma. An order picking heuristic algorithm for economical packing. In *Proceedings of the 11th IEEE International Conference on Networking, Sensing and Control*, pages 432–437. IEEE, apr 2014. ISBN 978-1-4799-3106-4.

[65] R. Storn and K. Price. Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4): 341–359, 1997.

[66] J. Tang, M. H. Lim, Y. S. Ong, and M. J. Er. Parallel memetic algorithm with selective local search for large scale quadratic assignment problems. *International Journal of Innovative Computing, Information and Control*, 2(6):1399–1416, 2006.

[67] C. Theys, O. Bräysy, W. Dullaert, and B. Raa. Using a TSP heuristic for routing order pickers in warehouses. *European Journal of Operational Research*, 200(3):755–763, 2010.

[68] J. A. Tompkins, J. A White, Y. A. Bozer, and J. M. A. Tanchoc. *Facilities planning.* J. Wiley, 2003. ISBN 0471413895.

[69] M. Veenstra, K. J. Roodbergen, I. F. A. Vis, and L. C. Coelho. The pickup and delivery traveling salesman problem with handling costs. *European Journal of Operational Research*, 257(1):118–132, 2017. ISSN 03772217.

[70] K. L. Yam. *The Wiley encyclopedia of packaging technology.* John Wiley & Sons, United States of America, 2009. ISBN 9780470087046.

[71] X.-S. Yang, editor. *Recent Advances in Swarm Intelligence and Evolutionary Computation*, volume 585 of *Studies in Computational Intelligence*. Springer International Publishing, 2015.

[72] X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi, and M. Karamanoglu, editors. *Swarm Intelligence and Bio-Inspired Computation. Theory and Applications*. Elsevier, 2013.

[73] J. Ye, M. Tanaka, and T. Tanino. Eugenics-based genetic algorithm. *IEICE Trans. on Information and Systems*, E79-D(5):600–607, 1996.

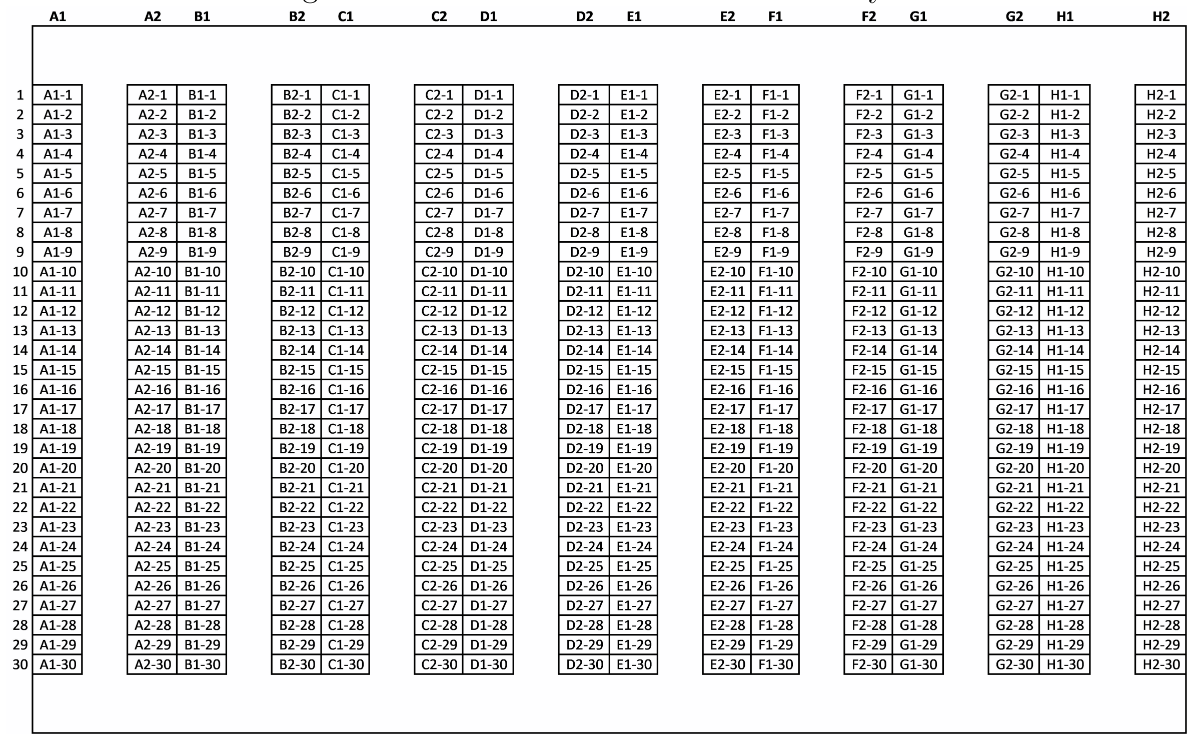
[74] D. Zhou, Y. Fang, J. Botzheim, N. Kubota, and H. Liu. Bacterial memetic algorithm based feature selection for surface emg based hand motion recognition in long-term use. In *Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7, Athens, Greece, December 2016.

Appendix

Layout definition

Figure 30 shows the Position ID definition logic and the layout. The clear areas are the aisles, the rectangles are the picking positions with their position ID.

Figure 30: Position ID definition and the layout



	A1	A2	B1	B2	C1	C2	D1	D2	E1	E2	F1	F2	G1	G2	H1	H2
1	A1-1	A2-1	B1-1	B2-1	C1-1	C2-1	D1-1	D2-1	E1-1	E2-1	F1-1	F2-1	G1-1	G2-1	H1-1	H2-1
2	A1-2	A2-2	B1-2	B2-2	C1-2	C2-2	D1-2	D2-2	E1-2	E2-2	F1-2	F2-2	G1-2	G2-2	H1-2	H2-2
3	A1-3	A2-3	B1-3	B2-3	C1-3	C2-3	D1-3	D2-3	E1-3	E2-3	F1-3	F2-3	G1-3	G2-3	H1-3	H2-3
4	A1-4	A2-4	B1-4	B2-4	C1-4	C2-4	D1-4	D2-4	E1-4	E2-4	F1-4	F2-4	G1-4	G2-4	H1-4	H2-4
5	A1-5	A2-5	B1-5	B2-5	C1-5	C2-5	D1-5	D2-5	E1-5	E2-5	F1-5	F2-5	G1-5	G2-5	H1-5	H2-5
6	A1-6	A2-6	B1-6	B2-6	C1-6	C2-6	D1-6	D2-6	E1-6	E2-6	F1-6	F2-6	G1-6	G2-6	H1-6	H2-6
7	A1-7	A2-7	B1-7	B2-7	C1-7	C2-7	D1-7	D2-7	E1-7	E2-7	F1-7	F2-7	G1-7	G2-7	H1-7	H2-7
8	A1-8	A2-8	B1-8	B2-8	C1-8	C2-8	D1-8	D2-8	E1-8	E2-8	F1-8	F2-8	G1-8	G2-8	H1-8	H2-8
9	A1-9	A2-9	B1-9	B2-9	C1-9	C2-9	D1-9	D2-9	E1-9	E2-9	F1-9	F2-9	G1-9	G2-9	H1-9	H2-9
10	A1-10	A2-10	B1-10	B2-10	C1-10	C2-10	D1-10	D2-10	E1-10	E2-10	F1-10	F2-10	G1-10	G2-10	H1-10	H2-10
11	A1-11	A2-11	B1-11	B2-11	C1-11	C2-11	D1-11	D2-11	E1-11	E2-11	F1-11	F2-11	G1-11	G2-11	H1-11	H2-11
12	A1-12	A2-12	B1-12	B2-12	C1-12	C2-12	D1-12	D2-12	E1-12	E2-12	F1-12	F2-12	G1-12	G2-12	H1-12	H2-12
13	A1-13	A2-13	B1-13	B2-13	C1-13	C2-13	D1-13	D2-13	E1-13	E2-13	F1-13	F2-13	G1-13	G2-13	H1-13	H2-13
14	A1-14	A2-14	B1-14	B2-14	C1-14	C2-14	D1-14	D2-14	E1-14	E2-14	F1-14	F2-14	G1-14	G2-14	H1-14	H2-14
15	A1-15	A2-15	B1-15	B2-15	C1-15	C2-15	D1-15	D2-15	E1-15	E2-15	F1-15	F2-15	G1-15	G2-15	H1-15	H2-15
16	A1-16	A2-16	B1-16	B2-16	C1-16	C2-16	D1-16	D2-16	E1-16	E2-16	F1-16	F2-16	G1-16	G2-16	H1-16	H2-16
17	A1-17	A2-17	B1-17	B2-17	C1-17	C2-17	D1-17	D2-17	E1-17	E2-17	F1-17	F2-17	G1-17	G2-17	H1-17	H2-17
18	A1-18	A2-18	B1-18	B2-18	C1-18	C2-18	D1-18	D2-18	E1-18	E2-18	F1-18	F2-18	G1-18	G2-18	H1-18	H2-18
19	A1-19	A2-19	B1-19	B2-19	C1-19	C2-19	D1-19	D2-19	E1-19	E2-19	F1-19	F2-19	G1-19	G2-19	H1-19	H2-19
20	A1-20	A2-20	B1-20	B2-20	C1-20	C2-20	D1-20	D2-20	E1-20	E2-20	F1-20	F2-20	G1-20	G2-20	H1-20	H2-20
21	A1-21	A2-21	B1-21	B2-21	C1-21	C2-21	D1-21	D2-21	E1-21	E2-21	F1-21	F2-21	G1-21	G2-21	H1-21	H2-21
22	A1-22	A2-22	B1-22	B2-22	C1-22	C2-22	D1-22	D2-22	E1-22	E2-22	F1-22	F2-22	G1-22	G2-22	H1-22	H2-22
23	A1-23	A2-23	B1-23	B2-23	C1-23	C2-23	D1-23	D2-23	E1-23	E2-23	F1-23	F2-23	G1-23	G2-23	H1-23	H2-23
24	A1-24	A2-24	B1-24	B2-24	C1-24	C2-24	D1-24	D2-24	E1-24	E2-24	F1-24	F2-24	G1-24	G2-24	H1-24	H2-24
25	A1-25	A2-25	B1-25	B2-25	C1-25	C2-25	D1-25	D2-25	E1-25	E2-25	F1-25	F2-25	G1-25	G2-25	H1-25	H2-25
26	A1-26	A2-26	B1-26	B2-26	C1-26	C2-26	D1-26	D2-26	E1-26	E2-26	F1-26	F2-26	G1-26	G2-26	H1-26	H2-26
27	A1-27	A2-27	B1-27	B2-27	C1-27	C2-27	D1-27	D2-27	E1-27	E2-27	F1-27	F2-27	G1-27	G2-27	H1-27	H2-27
28	A1-28	A2-28	B1-28	B2-28	C1-28	C2-28	D1-28	D2-28	E1-28	E2-28	F1-28	F2-28	G1-28	G2-28	H1-28	H2-28
29	A1-29	A2-29	B1-29	B2-29	C1-29	C2-29	D1-29	D2-29	E1-29	E2-29	F1-29	F2-29	G1-29	G2-29	H1-29	H2-29
30	A1-30	A2-30	B1-30	B2-30	C1-30	C2-30	D1-30	D2-30	E1-30	E2-30	F1-30	F2-30	G1-30	G2-30	H1-30	H2-30

Items and Storage Location Assignment (SLA)

Table 44 shows the ordered items of the O10, O20, and O50 orders. It describes the Product Class (PC) and the order picking position ID based on the SLA.

Table 44: Applied items and Storage Location Assignment

ItemID	PC	PosID
8	BigBox	A1-8
14	SmallBox	A1-14
15	Can	A1-15
23	Bag	A1-23
33	SmallBox	A2-3
38	SmallBox	A2-8
51	SmallBox	A2-21
66	BigBox	B1-6
77	Can	B1-17
78	Pail	B1-18
82	Pail	B1-22
90	BigBox	B1-30
92	SmallBox	B2-2
94	Pail	B2-4
100	Can	B2-10
110	Bag	B2-20
115	SmallBox	B2-25
122	Bag	C1-2
128	Can	C1-8
130	SmallBox	C1-10
149	Pail	C1-29
164	Pail	C2-14
171	Bag	C2-21
172	SmallBox	C2-22
173	Bag	C2-23
179	Can	C2-29
181	Can	D1-1
182	Fragile	D1-2
184	BigBox	D1-4
191	Bag	D1-11
192	SmallBox	D1-12
207	BigBox	D1-27
215	Pail	D2-5
219	Bag	D2-9
222	SmallBox	D2-12
223	Can	D2-13
225	SmallBox	D2-15
228	BigBox	D2-18
232	BigBox	D2-22
244	Can	E1-4
249	BigBox	E1-9
252	Pail	E1-12
266	Can	E1-26
267	SmallBox	E1-27
269	SmallBox	E1-29
271	SmallBox	E2-1
275	BigBox	E2-5
282	Pail	E2-12
288	BigBox	E2-18
291	Bag	E2-21
298	SmallBox	E2-28
302	SmallBox	F1-2
306	BigBox	F1-6
309	Bag	F1-9
313	BigBox	F1-13
314	SmallBox	F1-14
330	BigBox	F1-30
343	Can	F2-13
347	BigBox	F2-17
349	SmallBox	F2-19
354	BigBox	F2-24
360	BigBox	F2-30
371	Pail	G1-11
377	Fragile	G1-17
390	BigBox	G1-30
401	SmallBox	G2-11
418	Pail	G2-28
429	BigBox	H1-9
434	BigBox	H1-14
435	Pail	H1-15
438	BigBox	H1-18
449	Bag	H1-29
451	SmallBox	H2-1
455	BigBox	H2-5
459	Fragile	H2-9
461	SmallBox	H2-11
480	SmallBox	H2-30

Orders

Tables 45, 46, and 47 describe the order details of the applied O10, O20, and O50 orders.

Table 45: O10 order details

OrderItemID	Item	POPC	PosID
OI-1	15	Can	A1-15
OI-2	66	BigBox	B1-6
OI-3	115	SmallBox	B2-25
OI-4	172	SmallBox	C2-22
OI-5	223	Can	D2-13
OI-6	269	SmallBox	E1-29
OI-7	282	Pail	E2-12
OI-8	288	BigBox	E2-18
OI-9	349	SmallBox	F2-19
OI-10	360	BigBox	F2-30

Table 46: O20 order details

OrderItemID	Item	POPC	PosID
OI-1	271	SmallBox	E2-1
OI-2	267	SmallBox	E1-27
OI-3	23	Bag	A1-23
OI-4	184	BigBox	D1-4
OI-5	390	BigBox	G1-30
OI-6	354	BigBox	F2-24
OI-7	149	Pail	C1-29
OI-8	51	SmallBox	A2-21
OI-9	110	Bag	B2-20
OI-10	14	SmallBox	A1-14
OI-11	314	SmallBox	F1-14
OI-12	222	SmallBox	D2-12
OI-13	434	BigBox	H1-14
OI-14	459	Fragile	H2-9
OI-15	309	Bag	F1-9
OI-16	480	SmallBox	H2-30
OI-17	429	BigBox	H1-9
OI-18	343	Can	F2-13
OI-19	306	BigBox	F1-6
OI-20	330	BigBox	F1-30

Table 47: O50 order details

OrderItemID	Item	POPC	PosID
OI-1	249	BigBox	E1-9
OI-2	302	SmallBox	F1-2
OI-3	90	BigBox	B1-30
OI-4	94	Pail	B2-4
OI-5	191	Bag	D1-11
OI-6	100	Can	B2-10
OI-7	182	Fragile	D1-2
OI-8	23	Bag	A1-23
OI-9	418	Pail	G2-28
OI-10	266	Can	E1-26
OI-11	78	Pail	B1-18
OI-12	313	BigBox	F1-13
OI-13	225	SmallBox	D2-15
OI-14	275	BigBox	E2-5
OI-15	38	SmallBox	A2-8
OI-16	435	Pail	H1-15
OI-17	252	Pail	E1-12
OI-18	298	SmallBox	E2-28
OI-19	128	Can	C1-8
OI-20	244	Can	E1-4
OI-21	232	BigBox	D2-22
OI-22	77	Can	B1-17
OI-23	451	SmallBox	H2-1
OI-24	228	BigBox	D2-18
OI-25	33	SmallBox	A2-3
OI-26	349	SmallBox	F2-19
OI-27	438	BigBox	H1-18
OI-28	449	Bag	H1-29
OI-29	184	BigBox	D1-4
OI-30	219	Bag	D2-9
OI-31	192	SmallBox	D1-12
OI-32	371	Pail	G1-11
OI-33	181	Can	D1-1
OI-34	8	BigBox	A1-8
OI-35	130	SmallBox	C1-10
OI-36	455	BigBox	H2-5
OI-37	461	SmallBox	H2-11
OI-38	173	Bag	C2-23
OI-39	122	Bag	C1-2
OI-40	164	Pail	C2-14
OI-41	377	Fragile	G1-17
OI-42	179	Can	C2-29
OI-43	215	Pail	D2-5
OI-44	401	SmallBox	G2-11
OI-45	291	Bag	E2-21
OI-46	207	BigBox	D1-27
OI-47	347	BigBox	F2-17
OI-48	171	Bag	C2-21
OI-49	92	SmallBox	B2-2
OI-50	82	Pail	B1-22

Optimised sequences by StrictInit+NonStrictBM+BestDevLS-SA algorithm

Tables 48, 49, and 50 show the OrderItemID sequence (sequence of the order records) of the applied O10, O20, and O50 orders.

Table 48: The best solution for O10 list of the StrictInit+NonStrictBM+BestDevLS-SA algorithm

Sequence	OrderItemID
1.	OI-1
2.	OI-2
3.	OI-5
4.	OI-7
5.	OI-8
6.	OI-10
7.	OI-9
8.	OI-6
9.	OI-4
10.	OI-3

Table 49: The best solution for O20 list of the StrictInit+NonStrictBM+BestDevLS-SA algorithm

Sequence	OrderItemID
1.	OI-3
2.	OI-9
3.	OI-7
4.	OI-15
5.	OI-18
6.	OI-6
7.	OI-20
8.	OI-5
9.	OI-13
10.	OI-17
11.	OI-19
12.	OI-1
13.	OI-4
14.	OI-12
15.	OI-10
16.	OI-8
17.	OI-2
18.	OI-11
19.	OI-16
20.	OI-14

Table 50: The best solution for O50 list of the StrictInit+NonStrictBM+BestDevLS-SA algorithm

Sequence	OrderItemID
1.	OI-8
2.	OI-38
3.	OI-48
4.	OI-39
5.	OI-30
6.	OI-5
7.	OI-45
8.	OI-28
9.	OI-16
10.	OI-50
11.	OI-11
12.	OI-4
13.	OI-40
14.	OI-43
15.	OI-17
16.	OI-32
17.	OI-9
18.	OI-10
19.	OI-42
20.	OI-22
21.	OI-6
22.	OI-19
23.	OI-33
24.	OI-20
25.	OI-14
26.	OI-1
27.	OI-36
28.	OI-27
29.	OI-12
30.	OI-47
31.	OI-3
32.	OI-46
33.	OI-21
34.	OI-24
35.	OI-29
36.	OI-34
37.	OI-15
38.	OI-25
39.	OI-49
40.	OI-35
41.	OI-13
42.	OI-31
43.	OI-23
44.	OI-37
45.	OI-2
46.	OI-26
47.	OI-18
48.	OI-44
49.	OI-41
50.	OI-7