Journal of Manufacturing Systems 77 (2024) 384–397

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

journal homepage: www.elsevier.com/locate/jmansys

Technical paper

Integrated system configuration and layout planning for flexible manufacturing systems

Péter Dobrovoczki [a,b,*], András Kovács [a], Hiroyuki Sakata [c], Daisuke Tsutsumi [c]

[a] HUN-REN Institute for Computer Science and Control (SZTAKI), Kende u. 13-17, Budapest, 1111, Hungary
[b] Eötvös Loránd University, Pázmány Péter sétány 1/A, Budapest, 1117, Hungary
[c] Hitachi Ltd., Research & Development Group, 292 Yoshida-chyo, Totsuka-ku, Yokohama-shi, Kanagawa, 244-0817, Japan

ARTICLE INFO

Keywords:
System configuration
Layout planning
Flexible manufacturing systems
Combinatorial optimization
Logic-based benders decomposition

ABSTRACT

During the (re-)design of manufacturing systems, geometrical limitations on the available floor space may seriously impact the applicable resource configurations, including the selection of machines, robots, as well as auxiliary equipment. In current practice, such cases are managed by arduous manual iterations over the selection of resources and their geometrical arrangement. To overcome this inefficiency of existing approaches, the paper introduces a generic, integrated configuration-and-layout problem where the configuration sub-problem can encode arbitrary application-specific constraints on the selection of items (e.g., CNC machines and robots), while the layout sub-problem ensures geometrical feasibility, via a 2D rectangle packing representation. The generic model is demonstrated on an industrial application that involves the design of a flexible manufacturing system: items corresponding to CNC machines and robots must be selected, assigned to multiple manufacturing cells, and placed in the workshop blueprint to ensure that a given mix of products can be manufactured in the desired volume. For solving the generic configuration-and-layout problem, a logic-based Benders decomposition method is proposed. The efficiency of the approach is ensured by adding lifted cuts, symmetry breaking, and redundant constraints inspired by 2D bin packing lower bounds to the core Benders framework. Thorough computational evaluation is performed on a large set of problem instances, whereas practical applicability is verified in a real industrial case study.

1. Introduction

The design of complex engineering systems inevitably involves a wide range of elementary decisions, associated with constraints and objectives of different nature, which require different types of engineering knowledge, and accordingly, are made at different divisions of the enterprise along a complicated workflow. At the same time, these decisions are strongly interrelated, and selecting an attractive alternative at one decision step may easily result in sub-optimality or even infeasibility at a later phase of the workflow. In industrial practice, such cases are managed by iteratively revisiting the affected decisions until an acceptable compromise is attained.

In the (re-)design of manufacturing systems, it is typical to decompose the overall decision workflow into manufacturing system configuration (i.e., selecting the manufacturing resources and assigning production tasks to them), geometrical layout planning, motion planning and control design stages [1]. A plethora of different techniques is available in the scientific literature to tackle any of these planning levels *in itself*. Yet, according to this decomposition approach, system configuration (resource selection) is performed without considering geometric aspects. The selected configuration may turn out to be infeasible during geometrical layout planning if the selected equipment does not fit into the given floor space, or during motion planning if the selected single robot cannot serve all the machines in the workcell due to limited reach. In current practice, these cases are handled by tedious iterations between the different levels of planning. The iterations can be avoided only via tighter integration of the different planing levels. The current paper focuses on *integration* between manufacturing system configuration and layout planning. In order to tackle the above challenge, the paper introduces a generic, abstract model for integrated configuration-and-layout problems. The configuration sub-problem involves the optimal selection of items to be placed into containers (e.g., CNC machines and robots into cells in a manufacturing application), whereas the layout sub-problem is responsible for the precise geometrical arrangement of the selected items in the containers.

* Corresponding author at: HUN-REN Institute for Computer Science and Control (SZTAKI), Kende u. 13-17, Budapest, 1111, Hungary.
E-mail addresses: peter.dobrovoczki@sztaki.hu (P. Dobrovoczki), andras.kovacs@sztaki.hu (A. Kovács), hiroyuki.sakata.zy@hitachi.com (H. Sakata), daisuke.tsutsumi.jh@hitachi.com (D. Tsutsumi).

https://doi.org/10.1016/j.jmsy.2024.09.020
Received 29 January 2024; Received in revised form 11 September 2024; Accepted 28 September 2024
Available online 11 October 2024

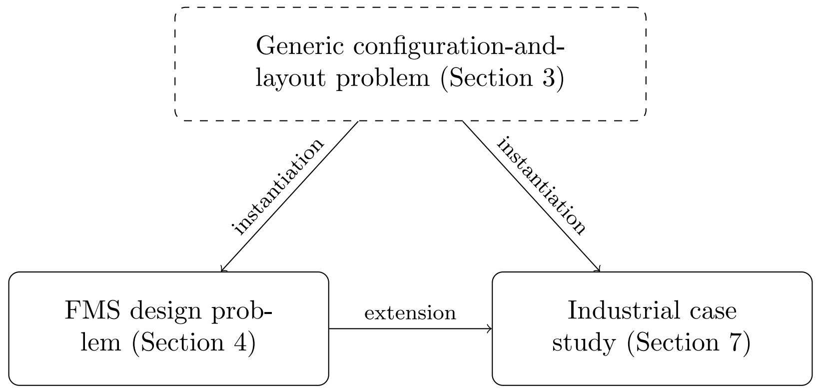

Fig. 1. Overview of the problem variants considered in this paper.

Given that both the configuration and the layout planning problems are NP-hard in themselves, solving the combined problem is particularly challenging. Such complex optimization problems with hierarchical structures can be efficiently addressed by a divide-and-conquer approach called *logic-based Benders decomposition* [2]: the overall problem is subdivided into an upper-level master problem (the configuration problem in this case) and a lower-level sub-problem (layout planning). During the search for an upper-level solution, each candidate solution is verified by solving the corresponding lower-level problem as well. Upon lower-level infeasibility, new constraints, so-called *Benders cuts* are fed back to the upper-level problem to eliminate similar sources of infeasibility in the future. This process is iterated until a joint optimal solution is found for both the upper- and the lower-level problems. This paper proposes a logic-based Benders decomposition to the above generic configuration-and-layout problem.

The paper investigates three variants of the problem as presented in Fig. 1. All solution techniques are developed for the *generic configuration-and-layout problem*. This is an abstract problem, without a precise definition of the decision variables and the constraints in the model, but only with the assumptions on the problem structure that are required for the solution approaches to work. This allows the application of the algorithms to specific problems in various, seemingly very different domains. In this paper, the solution techniques are demonstrated and evaluated on a specific *flexible manufacturing system* (FMS) design problem, which is an instantiation of the generic configuration-and-layout problem. Finally, the results from a real industrial case study are reported. The case study involves numerous practical extensions of the above FMS design problem model, and hence, it is another instantiation of the generic configuration-and-layout problem. Since these extensions are very specific to the given application, while they are less relevant for the research community, they are presented only in a brief textual description.

The main contributions of the paper are the following:

- A formal model and a Benders solution approach are defined for generic configuration-and-layout problems. The generic model comprises various applications, studied separately earlier in the literature, with different application-specific side constraints, as long as these side constraints satisfy some well-defined assumptions.
- The application of the generic configuration-and-layout approach in the manufacturing domain is illustrated on a specific FMS design problem.
- Various computational techniques, including new lifted cuts, symmetry breaking, and redundant constraints adapted from known 2D bin packing lower bounds are proposed for making the Benders approach computationally efficient.
- The methods are evaluated on a large set of FMS design problem instances that were generated randomly partly based on real industrial data.
- Results of a real industrial FMS design case study are reported.

The rest of the paper is structured as follows. First, a brief review of the related literature from manufacturing and operations research is given (Section 2). Then, the generic configuration-and-layout problem is defined (Section 3), and its application to FMS design is presented (Section 4). The proposed solution methods are introduced in Section 5 and evaluated in thorough computational experiments in Section 6. The lessons learnt from the real industrial case study are summarized in Section 7. Finally, conclusions are drawn and directions for future research are proposed in Section 8.

2. Literature review

2.1. Configuration and layout planning in manufacturing

Production system configuration addresses finding the optimal combination of resources for manufacturing a product or a family of products in the requested volume. A recent review of design approaches for manufacturing systems according to different flexibility paradigms, including cellular, flexible, and reconfigurable systems is given in [3]. The scientific literature on production system configuration can be divided into two substantially different approaches: (1) the sophisticated, multi-criteria evaluation and comparison of candidate configurations given in the input, usually designed by human engineers, and (2) the automated synthesis of the optimal configuration for well-defined requirements. The former direction is particularly relevant in applications where it is difficult to set up a clear-cut, deterministic optimization model, typically due to severe uncertainties related to product life cycles, demand volumes, or manufacturing processes. Contributions include techniques for evaluating complete system configurations, such as [4], where the evaluation and filtering of the candidate configurations of a complex FMS are performed on different levels of hierarchy using a combination of analytic methods and simulation; [5], where a fuzzy Analytic Hierarchy Process (AHP) approach is proposed for evaluating candidate configurations based on a large number of criteria; as well as papers focusing on the selection of a single resource at a time, such as [6], which addresses choosing the most suitable collaborative robot for a application using a hybrid AHP-TOPSIS methodology.

In the sequel, we focus on the second direction, i.e., mathematical optimization approaches to the automated synthesis of system configurations from well-defined requirements. These approaches take as input the specification of the product mix, the corresponding process plans, as well as the forecast demand volumes, and ask for the combination of resources from a given resource library that can produce the requested product mix in the most efficient way, usually, with the lowest possible investment cost. The elementary decisions to make depend greatly on the architecture of the production system and the model features. Many contributions formulate optimization models heavily tailored to the requirements of a specific application: [7] captures the configuration problem of flow lines composed of reconfigurable machines for batch production in the form of a mixed-integer linear programming (MILP) model; [8] addresses system configuration for fully exploiting the potential of co-platforming via optimal mapping between product platforms and machine platforms, and again formulates the problem as a MILP; [9] decomposes the overall configuration problem of an automotive Li-ion battery pack assembly line into task grouping, sequence planning, and equipment selection sub-problems, and apply custom enumeration methods with recourse between the sub-problems to solve it.

Beyond these highly application-specific models, well-established families of optimization models and solution approaches are available for configuration problems arising in certain system architectures, such as flow systems and cellular manufacturing systems. For flow systems, classical optimization approaches rely on different variants and extensions of the *assembly line balancing problem*. Models and classification schemes are discussed in [10,11], whereas an excellent survey on recent developments is presented in [12]. Typical solution methods include mathematical programming, custom branch-and-bound algorithms, and (meta-)heuristics [13,14]. Benders decomposition has been applied to solving assembly line balancing with sequence-dependent setups in [15] by decomposing the problem into an assignment master problem and a sequencing sub-problem. Further applications of the Benders approach to richer models include line balancing and sequencing subject to stochastic demand [16], line balancing with walking workers [17], as well as balancing multi-manned stations [18]. Line balancing is more and more frequently solved in combination with other related planning problems, including process planning, task sequencing [19], the detailed configuration of resources at individual stations, or detailed scheduling [20]. The problem of adapting an existing assembly line to meet altered requirements, i.e., the industrially relevant *brownfield* problem instead of the most commonly studied *greenfield* problem, is investigated in [21].

A strongly related problem in cellular manufacturing systems is called *cell formation* [22]. In the basic version of this problem, a given set of machines must be assigned to cells in the manufacturing system in order to minimize logistic cost or effort. This is solved often in combination with the intra-cell layout problem, looking for an optimal placement of the machines within a cell, or the inter-cell layout problem, responsible for locating the cells within the plant [23]. Since the combined problem is computationally challenging, it is solved typically using meta-heuristic approaches, such as simulated annealing [24], tabu search [25], or a genetic algorithm [26]. However, most of these approaches make simplifying assumptions compared to the problem investigated in this paper: the set of machines is fixed (hence, no investment cost is considered), and geometry is simplified to uniform slots [23,24], one-dimensional sections in a single-row [25], double-row [27], or multi-row arrangement [26]. Some contributions assume that items can be placed arbitrarily in the continuous 2D space, resulting in a so-called *open-field* layout [28]. The choice of the geometrical model depends primarily on the selected material handling system, since a single-row layout can be efficiently operated using a cheap conveyor belt, whereas in flexible manufacturing systems with automated parts handling using automated guided vehicles (AGVs), an open-field layout can lead to significantly lower travel distances and material handling costs. A common characteristic of the above layout planning approaches is the application of abstract geometry in order to ensure a rough geometrical feasibility and to optimize the chosen logistic objective.

Obviously, in various delicate applications, such as the design of robotic assembly cells, the detailed layout must be planned meticulously before building the cell in physical reality, based on precise geometric and kinematic models. Such approaches include [29], where the optimal robot base placement in a welding cell is determined considering the given end-effector trajectory, robot kinematics and reachability, as well as collision avoidance. [30] investigates optimal workpiece placement in the robot workspace in additive manufacturing to maximize the geometrical precision of the product and to minimize the energy consumption of the manufacturing process. Nevertheless, working with such detailed geometries and motion plans is clearly out of scope in the early system configuration stage considered in the current paper.

2.2. Layout planning as a packing problem in operations research

Since the open-field layout planning sub-problem of the model investigated in this paper corresponds to a rectangle packing problem, exact solution methods for the latter are of particular interest. A recent survey of such exact methods, covering mathematical programming approaches, enumeration techniques, as well as relaxations and heuristics, is presented in [31], whereas an earlier, classical review is available in [32]. Lower bounds and a heuristic for 2D bin packing where a subset of the items can be rotated by 90° are proposed in [33].

Logic-based Benders decomposition has been applied to packing problems in various ways. [34] and [35] propose two similar exact solution methods using Benders decomposition for 2D orthogonal cutting stock problems, without and with rotation, respectively. The upper level looks for x coordinates, whereas the lower level assigns y coordinates. Various techniques to strengthen the Benders cuts are introduced, including lifting techniques and the heuristic identification of a minimal infeasible set of items. Pre-processing techniques that adjust item and container sizes to tighten relaxations, applicable to rectangle packing with 90° rotation, are also presented in [35].

In some applications, the packing problem is combined with other higher-level problems. A review of routing problems with loading constraints is presented in [36]. For a variant of this problem, [37] proposes a branch-and-cut approach where an embedded branch-and-bound algorithm checks the feasibility of the involved loading sub-problems. Integrated lot-sizing and cutting stock problems are surveyed in [38]. The reviewed papers differ in the number of levels in the inventory management sub-problem, as well as the involved cutting stock (equivalently, packing) sub-problem. For the latter, one-dimensional problems are typical, but two- or three-dimensional variants are also identified. In [39], feasible cutting patterns are generated and then a Lagrangian-based heuristic is applied for computing a close-to-optimal solution for the integrated problem. [40] and [41] address the problem using column generation approaches. [42] introduces a heuristic method motivated by the network shortest path problem. The generation of cutting patterns, i.e., the pricing sub-problem is solved using the classical method of Gilmore and Gomory [43] for 2-stage guillotine cuts.

While the above problems fit into the generic model introduced in this paper, the computational challenge is somewhat different: in the above routing and inventory problems, the set of items is fixed; they must be assigned to containers according to very simple rules; yet, the emphasis is on solving large-size problems efficiently. In contrast, in the manufacturing system configuration problem investigated here, the items (machines and robots) must be selected subject to a set of complex constraints (i.e., the upper-level configuration problem is computationally challenging in itself), while real industrial applications can be covered by solving moderate-size problems.

2.3. Positioning of the paper

The above literature review showed that both manufacturing system configuration and layout planning have received significant attention in the scientific literature *in themselves*, separately. Yet, there are hardly any results available on the *integration* of these two interrelated problems. There exist models that can be positioned on the borderline between these two levels of the planning hierarchy, e.g., integrated cell formation and layout planning in cellular manufacturing. Yet, the scope of these models is different, in a way narrower than what is addressed in this paper: they do not consider resource selection, perform layout planning using an abstract representation (e.g., uniform slots or linear layout), and they focus on minimizing internal logistic effort, rather than investment cost. Overall, there is a lack of well-established methods in the literature for integrating manufacturing system configuration and layout planning.

Logic-based Benders decomposition is a typical divide-and-conquer strategy for solving combinatorial optimization problems that can be separated into loosely coupled upper- and lower-level sub-problems. Yet, in applications where both the upper- and the lower-level sub-problems are difficult, i.e., NP-hard, achieving computational efficiency is still a challenge. Developing problem-specific inference techniques, such as lifted cuts or algorithms for identifying minimal infeasible sets, can boost the performance of the approach.

The current paper takes a step beyond the state-of-the-art in two different ways. In terms of modeling, it defines a generic configuration-and-layout problem model; the generic model integrates the system configuration and the layout planning aspects of manufacturing system design into a single problem model, while it allows adding application-specific side constraints as long as these satisfy some well-defined assumptions. In terms of algorithms, the paper proposes a logic-based Benders decomposition approach to the generic configuration-and-layout problem, and defines various computational techniques, namely, symmetry breaking, lifted cuts and redundant constraint, that help achieve computational efficiency. The effectiveness and efficiency of the proposed approach is illustrated both in computational experiments on a large set of problem instances and in a real industrial case study involving engineers of the industrial partner.

3. The generic configuration-and-layout problem

The generic *configuration-and-layout problem* addresses selecting items from a predefined set, assigning each selected item to one of the multiple containers according to application-specific constraints and objective (configuration sub-problem), and then geometrically arranging the assigned items in each container (layout sub-problem). The geometry of the items and the containers, as well as the number of available containers is given in the input. The assignment between items and containers is modeled via so-called *slots*. A slot is an abstract position within a container that can host at most one item. Slots do not have any geometrical meaning: they do not have predetermined sizes or geometrical positions within the container. Yet, the number of slots within a container is given in the input, and this implies an upper bound on the number of items that can be assigned to the container. Both the configuration and the layout sub-problems are defined formally in the following sections, whereas a specific application of the abstract model is presented in Section 4.

3.1. Configuration sub-problem

The generic configuration sub-problem is defined as follows. Let $u_{c,s,i}$ denote the binary *selection variable* whose value is 1 if and only if item i is selected to slot s of container c. For the sake of generality, the model allows a set of additional variables v, where J denotes the index set of these variables, and set $I \subseteq J$ contains the indices of integer variables. In order to encode arbitrary application-specific constraints and objective, let $A \in \mathbb{R}^{n_1 \times m}, B \in \mathbb{R}^{n_2 \times m}$ be the coefficient matrices of appropriate dimension, $b \in \mathbb{R}^m$ be a bounding vector and $c \in \mathbb{R}^{n_1}, d \in \mathbb{R}^{n_2}$ be components the objective function. Let N denote the number of items, S denote the number of available slots in a container and C denote the number of containers. With this, the following MILP formulates the abstract configuration sub-problem:

$$\text{minimize} \quad cu + dv \tag{1a}$$

$$\text{subject to} \quad Au + Bv \le b \tag{1b}$$

$$v_j \in \mathbb{Z} \qquad \forall j \in I \tag{1c}$$

$$u \in \{0,1\}^{C \times S \times N} \tag{1d}$$

3.2. Layout sub-problem

The layout sub-problem addresses the geometrical arrangement of the selected items in their assigned containers. As above, an abstract definition of the layout sub-problem is given first with the assumptions that are required for the proposed solution techniques to work, whereas specific instantiations of the generic model will be presented in detail later.

- It is assumed that both containers and items are rectangular, and their dimensions are known a priori.
- Rotation of the items by 90° is allowed (although the approach can trivially be adapted to the simpler case where rotation is denied).
- All items assigned to a container must be placed fully within the container, and they must not overlap each other.
- No constraints connect the layout sub-problems related to different containers, and accordingly, layout planning can be solved separately for each container.
- Layout planning is a *feasibility problem*, i.e., its solution does not impact the objective value of the overall configuration-and-layout problem.
- The layout sub-problem is monotonous in the sense that if a feasible layout exists in container c for a set of items I_c, then a feasible layout exists for subset $I'_c \subset I_c$ as well.

In specific applications, this abstract model can be extended with arbitrary side constraints as long as they do not violate the above assumption. A simple, specific instantiation of the abstract layout sub-problem is presented in Section 4.3, whereas a richer, industrial application with various side constraints is discussed in Section 7.

4. Application to flexible manufacturing systems

This section presents how the generic configuration-and-layout model can be applied to capturing a challenging industrial FMS design problem. For this purpose, detailed definitions of the configuration and layout sub-problems are given, accompanied by the corresponding MILP formulations.

4.1. The FMS configuration-and-layout problem

The application involves the configuration of an FMS composed of multiple manufacturing cells to produce multiple products. Each cell contains two types of items: multiple *machines* for executing the single required manufacturing task on each product, and a *robot* for feeding the products into the machines before, as well as removing the product after the manufacturing task. Both machines and robots are available in different types, which are characterized by different capabilities (e.g., product dimensions imply constraints on the applicable machines and robots), speeds, and costs.

The forecast demand D_p per month is known in advance for each product p, and exactly this amount must be produced during the given planning horizon. The production of the total demand of a product can be divided among different cells and different machines in those cells. The cells of the system can differ arbitrarily from each other, and also, different types of machines can be deployed into the same cell.

Manufacturing one unit of a product p by robot type j requires a given *handling time* $T^H_{p,j}$ before and after processing the product, which occupies both the machine and the robot. Moreover, machining the same product by machine type i requires a *process time* $T^P_{p,i}$, during which only the machine is employed. If a machine or a robot is not suitable for a product, then this is captured by $T^H_{p,j} = \infty$ and $T^P_{p,i} = \infty$,

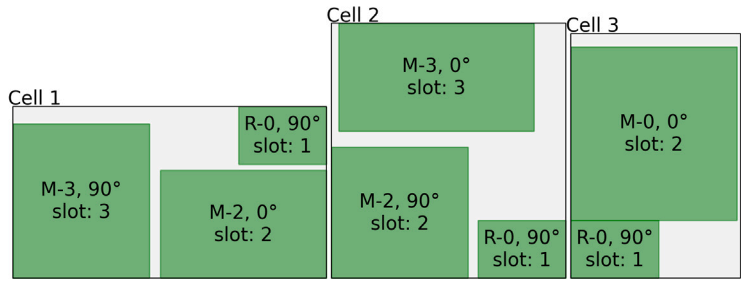

Fig. 2. Sample FMS configuration and layout with three cells. The label of the leftmost green box indicates that an instance of Machine type 3 is placed with 90° rotation into Slot 3 of Cell 1. Items labeled R represent robots. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

respectively. Although multiple machines within a cell compete for the same robot, the current model assumes that blockage times are negligible.

The available floor space is also limited, and the floor space requirement of each machine and robot is modeled as an axis-aligned rectangle. Namely, a robot of type j is described as a rectangle of size $w_{R(j)} \times h_{R(j)}$, whereas a machine of type n by a rectangle of size $w_{M(n)} \times h_{M(n)}$. Cell boundaries are determined by the shop floor architecture, resulting in a $W_c \times H_c$ rectangular space for each cell c.

The cost of operating a robot of type j over the monthly horizon is denoted by K^R_j, whereas for a machine of type i by K^M_i, which includes the depreciation, operation and maintenance costs. Then, the objective is minimizing the total cost. An example of an FMS configuration and layout with three cells is presented in Fig. 2. Each cell contains a robot for material handling (rectangles labeled R) and one or two machines (labels M). The labels also indicated whether an item is rotated (90°) or not (0°).

Observe that the specific FMS design problem fits into the generic configuration-and-layout model as follows. Manufacturing cells correspond to *containers*. The robots and machines are the *items* that must be selected and placed in the containers. In particular, one robot and at most $S-1$ machines must be selected for each cell, where S is given in the input. Accordingly, there are S slots in each cell, and the model assumes that the robot is assigned to slot 1, whereas machines to slots $2,\ldots,S$. Abstract assignment to slots does not imply any constraint on geometrical placement. It is also possible to leave a cell empty, if the remaining cells are sufficient to satisfy all demand. Then, all complex constraints that define how the system must be configured to produce the different products in the desired volume can be encoded in constraint (1b) of the generic configuration model. The details of the encoding are presented in the next section, whereas the notation for the FMS configuration sub-problem is summarized in Table 1.

4.2. MILP model for the FMS configuration sub-problem

The configuration sub-problem for the specific FMS design application can be formulated by the following MILP:

Minimize

$$\sum_{c,j} K^R_j u_{c,1,j} + \sum_{c,s,i\,:\,s\geq 2} K^M_i u_{c,s,i} \tag{2a}$$

subject to

$$\sum_{c,s,i,j\,:\,s\geq 2} v_{p,c,s,i,j} = 1 \qquad \forall p \tag{2b}$$

$$\sum_{s,i\,:\,s\geq 2} v_{p,c,s,i,j} \leq u_{c,1,j} \qquad \forall p,c,j \tag{2c}$$

$$\sum_{j} v_{p,c,s,i,j} \leq u_{c,s,i} \qquad \forall p,c,s,i : s\geq 2 \tag{2d}$$

$$\sum_{j} u_{c,1,j} \leq 1 \qquad \forall c \tag{2e}$$

$$\sum_{i} u_{c,s,i} \leq \sum_{j} u_{c,1,j} \qquad \forall c,s : s\geq 2 \tag{2f}$$

Table 1
Notation for the FMS configuration sub-problem.

Indices	
p	Product (index)
c	Cell (index)
s	Slot within cell (index)
i	Machine type (index)
j	Robot type (index)
Input parameters	
D_p	Demand for product p [pcs]
Θ	Length of the time horizon [min]
$T^P_{p,i}$	Process time for product p with machine type i [min]
$T^H_{p,j}$	Handling time for product p with robot type j [min]
$w_{R(j)}$, $h_{R(j)}$	Size of robot type j in dimensions x and y [m]
$w_{M(i)}$, $h_{M(i)}$	Size of machine type i in dimensions x and y [m]
W_c, H_c	Size of cell c in dimensions x and y [m]
K^M_i	Cost associated to machine type i [yen]
K^R_j	Cost associated to robot type j [yen]
Decision variables	
$u_{c,1,j}$	Cell c is built to operate with robot type j (binary)
$u_{c,s,i}$	In cell c, machine type i is assigned to slot s with $s\geq 2$ (binary)
$v_{p,c,s,i,j}$	Fraction of demand for product p assigned to cell c, slot s, to be produced with machine type i and robot type j (real in $[0,1]$)

$$\sum_{p,i,j} (T^P_{p,i} + T^H_{p,j}) D_p v_{p,c,s,i,j} \leq \Theta \sum_{i} u_{c,s,i} \qquad \forall c,s : s\geq 2 \tag{2g}$$

$$\sum_{p,s,i,j} T^H_{p,j} D_p v_{p,c,s,i,j} \leq \Theta \sum_{j} u_{c,1,j} \qquad \forall c \tag{2h}$$

$$u_{c,1,j},\ u_{c,s,i} \in \{0,1\} \qquad \forall c,s,i,j : s\geq 2 \tag{2i}$$

$$v_{p,c,s,i,j} \geq 0 \qquad \forall p,c,s,i,j : s\geq 2. \tag{2j}$$

The objective is minimizing the total cost (2a), where slot $s=1$ is reserved for the robot and slots $s\geq 2$ are for the machines. Constraint (2b) ensures that all demand is fully satisfied. Inequalities (2c) and (2d) state that a product can be assigned to a given machine and robot type only if those resources are available in the cell. Line (2e) states that a cell cannot be built with multiple robots. Constraint (2f) encodes that at most one machine can be built into each slot within a cell, and only if the cell itself contains a robot. Lines (2g) and (2h) capture the capacity constraints on the machines and the robots, respectively. Finally, the binary and the non-negative variables are enumerated in constraints (2i) and (2j).

4.3. MILP model for the FMS layout sub-problem

The FMS layout sub-problem is a particular instantiation of the generic layout planning sub-problem. Items must be placed entirely within the container and they must not overlap each other, while there are no further side constraints defined on the layout. This layout planning problem can be solved by any exact algorithm. In this paper, a MILP solution approach is chosen, since the number of items is small, and the computational cost of finding a feasible solution using MILP is

Table 2
Notation for the FMS layout sub-problem.

Indices and dimensions	
i	Item (index)
c	Cell (index)
Input parameters	
I_c	Items assigned to cell c (set)
w_i	Width of item i (real) [m]
h_i	Height of item i (real) [m]
W_c	Width of cell c (real) [m]
H_c	Height of cell c (real) [m]
Decision variables	
x_i	Coordinate x of the midpoint of item i (real) [m]
y_i	Coordinate y of the midpoint of item i (real) [m]
α_{ij}	Denotes if item i is above item j (binary)
ρ_{ij}	Denotes if item i is on the right of item j (binary)
R_i	Denotes if item i is rotated (binary)

insignificant compared to solving the upper-level problem. The notation for the layout sub-problem is shown in Table 2.

The following MILP formulation characterizes the feasible layouts in cell c with respect to the requirements and assumptions listed above.

$$x_i \geq \frac{\left(1-R_i\right) w_i + R_i h_i}{2} \quad \forall i \in \mathcal{I}_c \tag{3a}$$

$$x_i \leq W_c - \frac{\left(1-R_i\right) w_i + R_i h_i}{2} \quad \forall i \in \mathcal{I}_c \tag{3b}$$

$$y_i \geq \frac{\left(1-R_i\right) h_i + R_i w_i}{2} \quad \forall i \in \mathcal{I}_c \tag{3c}$$

$$y_i \leq H_c - \frac{\left(1-R_i\right) h_i + R_i w_i}{2} \quad \forall i \in \mathcal{I}_c \tag{3d}$$

$$\alpha_{i,j} \leq 1-\alpha_{j,i} \quad \forall i \neq j \in \mathcal{I}_c \tag{3e}$$

$$\rho_{i,j} \leq 1-\rho_{j,i} \quad \forall i \neq j \in \mathcal{I}_c \tag{3f}$$

$$\alpha_{i,j}+\rho_{i,j} \geq 1-\alpha_{j,i}-\rho_{j,i} \quad \forall i \neq j \in \mathcal{I}_c \tag{3g}$$

$$\left(\alpha_{i,j}=1\right) \implies y_i - \frac{(1-R_i)h_i + R_i w_i}{2} \geq y_j + \frac{(1-R_j)h_j + R_j w_j}{2} \quad \forall i \neq j \in \mathcal{I}_c \tag{3h}$$

$$\left(\rho_{i,j}=1\right) \implies x_i - \frac{(1-R_i)w_i + R_i h_i}{2} \geq x_j + \frac{(1-R_j)w_j + R_j h_j}{2} \quad \forall i \neq j \in \mathcal{I}_c \tag{3i}$$

$$\alpha_{i,j}, \rho_{i,j} \in \{0,1\} \quad \forall i \neq j \in \mathcal{I}_c \tag{3j}$$

$$R_i \in \{0,1\} \quad \forall i \in \mathcal{I}_c \tag{3k}$$

The MILP verifies the existence of a feasible layout, and accordingly, the objective function is not defined (or equivalently, identically zero in an implementation). Inequalities (3a)–(3d) are the constraints on the absolute positions of the items assigned to cell c: e.g., inequality (3a) ensures that the x coordinate of the midpoint of item i is at least as large as the half of the width of item i (or if item i is rotated and $R_i = 1$, then at least as large as the half of the height of item i). Constraint (3e) (respectively, (3f)) encodes that item i cannot be both above and under (both on the left and on the right of) item j. Inequality (3g) encodes that for relative position of item i and item j at least one of the following should be true: i is above j, i is below j, i is on the right of j, i is on the left of j. Indicator constraint (3h) encodes that if item i is above item j, then the y coordinate of the midpoint of item i should be greater than or equal to the y coordinate of the midpoint of item j plus half of their widths or heights, depending on whether they are rotated or not, respectively. Line (3i) states the same for the x coordinates of items located on the left or right of each other. Constraints (3j) and (3k) ensure that the relative positional variables and the rotational variables are binary.

5. Solution method

A major challenge in solving the integrated configuration-and-layout problem lies in expressing that a feasible layout must exist without introducing an excessive amount of variables and constraints. To overcome this challenge, logic-based Benders decomposition is applied to separate the overall problem into an upper-level configuration sub-problem (also called Benders master problem), and a lower-level layout sub-problem. The layout sub-problem iteratively generates cuts to the master problem that eliminate assignments that do not admit a feasible layout, as shown in Fig. 3.

The Benders approach is implemented using a branch-and-cut algorithm as follows. During the search for the optimal solution of the upper-level configuration sub-problem, the LP relaxation of this problem is solved in each node of the search tree. If the LP solution is integer, i.e., it is a feasible solution to the configuration sub-problem, then the corresponding layout sub-problem is solved as well. If there exists a feasible layout for the given configuration, then this solution is stored. Otherwise, the configuration is rejected, and Benders cuts are added to the upper-level configuration sub-problem to prevent the repeated occurrence of a similar kind of infeasibility. The approach guarantees finding an exact optimal solution given that sufficient computation time is available. If, for a challenging problem instance, the algorithm hits the time limit after finding a feasible solution, then the algorithm returns that possibly sub-optimal solution and the associated optimality gap, i.e., a bound on the distance from the exact optimal solution. The procedure executed in each node of the branch-and-cut search tree for solving the upper-level configuration problem is depicted in Fig. 4. The details of the algorithm, including the Benders cuts and the techniques to strengthen the formulation of the upper-level configuration sub-problem are presented in this section.

5.1. Benders cuts

For each container that does not admit a feasible layout, the procedure generates a new constraint (or possibly, multiple new constraints) to be added to the configuration master problem that exclude assigning the same, infeasible subset of items to the same container in future iterations.

5.1.1. No-good cuts

The classical no-good Benders cut excludes the assignment of a specific, infeasible subset of items to a container. Let c denote a container that admits no feasible layout, $\mathcal{I}_c = \left\{i_1, i_2, \ldots, i_k\right\}$ denote the k items assigned to container c. Let n_j stand for the type of item i_j, and s_j denote the slot where item i_j is assigned. Then, by exploiting the monotonicity of the layout problem and the fact that at most one item is assigned to each slot, the configuration sub-problem can be extended by the following no-good cut:

$$\sum_{j=1}^{k} u_{c,s_j,n_j} \leq k-1 \tag{4}$$

5.1.2. Lifted cuts

While the classical no-good cut excludes the assignment of a single, infeasible assignment, it can be strengthened by exploiting dominance relations to further assignments whose infeasibility can be inferred. For that purpose, a partial ordering over the set of items, as well as over the set of containers is defined as follows. Let $i \geq i'$ denote that in any feasible container layout, item i is exchangeable to item i'. Analogously for containers, let $c \preceq c'$ denote that if there exists a feasible layout for any given subset of items I_c in (the smaller) container c, then the same subset of items I_c can also be arranged feasibly in (the larger) container c'.

Obviously, the precise definition of the partial orders depend on the specific layout planning model. In the FMS design application, the

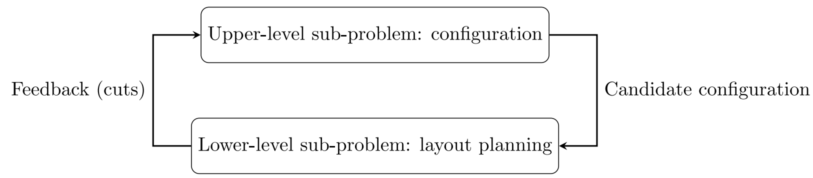

Fig. 3. Underlying idea of logic-based Benders decomposition.

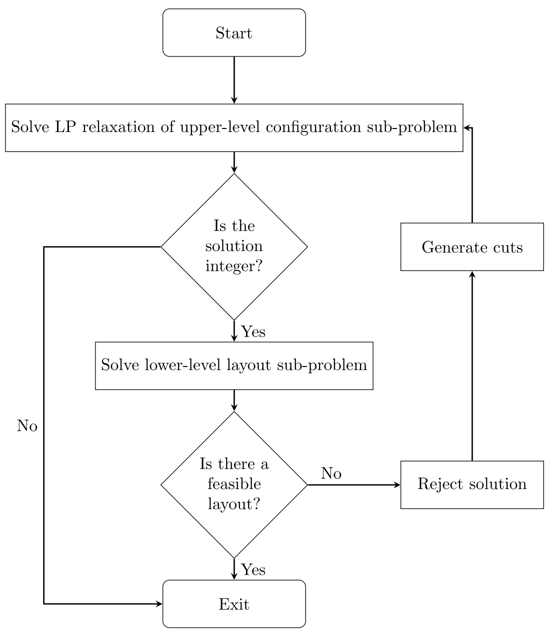

Fig. 4. The method executed in each node of the branch-and-cut search tree during the solution of upper-level configuration sub-problem for implementing the Benders approach.

layout sub-problem for each individual container corresponds to rectangle packing with 90° rotation, and therefore $i \geq i'$ if $\max(w_i, h_i) \geq \max(w_{i'}, h_{i'})$ and $\min(w_i, h_i) \geq \min(w_{i'}, h_{i'})$. For containers, the partial ordering is defined similarly.

The *internal dominance rule* captures that if a subset of items does not fit into container c, then neither do larger items. We can use the partial ordering of items to derive the lifted cut

$$\sum_{j=1}^{k} \sum_{\ell\,:\,n_\ell \geq n_j} u_{c,s_j,n_\ell} \leq k-1. \tag{5}$$

Likewise for containers, if container c does not admit a feasible layout for a subset of items, then neither does container c' with $c' \leq c$. Hence, we can apply the *external dominance rule* to derive the stronger, lifted cut:

$$\sum_{j=1}^{k} u_{c',s_j,n_j} \leq k-1 \quad \forall c' \in C : c' \leq c. \tag{6}$$

Combining these two ideas we get a set of new cuts:

$$\sum_{j=1}^{k} \sum_{\ell\,:\,n_\ell \geq n_j} u_{c',s_j,n_\ell} \leq k-1 \quad \forall c' \in C : c' \leq c. \tag{7}$$

Observe that for $c' = c$, inequality (7) dominates the classical no-good cut (4), while for $c' \neq c$, it is a valid inequality for all integer solutions of Problem (1) that admit a feasible layout.

5.2. Symmetry breaking

Let $S_1, S_2, \ldots, S_\ell$ be a partitioning of the slots such that any two slots within a partition class can be interchanged without effecting the feasibility and the objective value of the solution. Formally, a partitioning of the slots $S_1, S_2, \ldots, S_\ell$ is valid if, for any $c = 1, \ldots, C$, $i = 1, \ldots, N$, $j = 1, \ldots, \ell$ and $s, s' \in S_j$, the two assignments u and u' either both induce a feasible solution with identical objective values or they are both infeasible, where $u'_{c,s,i} = u_{c,s',i}$ and $u'_{c,s',i} = u_{c,s,i}$, and for all else indices $u' = u$. If such a partitioning exists, we call the problem symmetric in the slots, and symmetry breaking rules can be applied.

Assuming that the configuration-and-layout problem is symmetric in the slots, the formulation can be strengthened via symmetry breaking constraints. Symmetries can be broken by filling up the slots within each partition in an increasing order, i.e., for all $c \in [C]$, if there exists no $i \in [N]$ that $u_{c,s,i} = 1$, then for all $i' \in [N]$ we have $u_{c,s',i'} = 0$ for all $s' > s$ where $s, s' \in S_j$ for some j. This property can be ensured by adding the following constraints to Problem (1):

$$\sum_{i=1}^{N} u_{c,s,i} \geq \sum_{i=1}^{N} u_{c,s+1,i} \quad \forall c = 1, \ldots, C,\ \forall j = 1, \ldots, \ell,\ \forall s \in S_j. \tag{8}$$

The formulation can be strengthen further by prescribing that beside filling up the slots in increasing order, the items are also in a non-decreasing order of their indices within a container, i.e., if $u_{c,s,i} = 1$, then for all $j = 1, \ldots, \ell$, $s' > s$, $i' < i$ where $s, s' \in S_j$ we need $u_{c,s',i'} = 0$. This can be enforced by the following constraint:

$$\sum_{k=1}^{i} u_{c,s,k} \quad \geq \quad u_{c,s+1,i} \qquad \forall c = 1, \ldots, C,\ \forall i = 1, \ldots, N, \qquad \forall j = 1, \ldots, \ell,\ \forall s \in S_j. \tag{9}$$

Observe that the set of inequalities (9) dominate the set of inequalities (8). Furthermore, if Problem (1) is symmetric in the slots, extending its formulation with inequalities (9) yields a problem that has the same objective value but has a significantly smaller set of feasible solutions.

In case of the FMS application above, two partitions S_1, S_2 of the slots are created, with $S_1 = \{1\}$ corresponding to the single robot slot, and $S_2 = 2, \ldots, S$ to the machine slots. Symmetry breaking is applied to these two partitions.

5.3. Redundant constraints from geometry

The formulation of the upper-level problem can be strengthened further with constraints that arise from the geometric nature of the lower-level problem.

5.3.1. Mutually exclusive items

Items i and j are called mutually exclusive with respect to container c if and only if they cannot be placed into container c at the same time. Taking into consideration possible 90° rotations, this property can be checked with 8 comparisons per pair of items and containers. If items i and j are mutually exclusive w.r.t. container c, then inequalities

$$u_{c,s',i} + u_{c,s'',j} \leq 1 \quad \forall s' \neq s'' \in [S] \tag{10}$$

are all valid for the upper-level problem. Let $\mathcal{X}_c$ denote the set of mutually exclusive pairs for container c. Using that notion, inequalities (10) yield $\sum_{c\in[C]} |\mathcal{X}_c| \times \binom{S}{2}$ additional constraints. Although it is possible to extend the concept of exclusivity to more than two items, the number of additional constraints can grow large quickly.

5.3.2. Redundant constraints inspired by 2D bin packing lower bounds

Another possible way to leverage the geometric aspects of the lower-level problem is to use lower bounds of two-dimensional bin packing problems to get valid inequalities for the upper-level problem. Boschetti and Mingozzi propose lower bounds for the oriented case of two-dimensional bin packing problem in [44] and a simple transformation of the inputs to turn them into lower bounds for the rotational case in [33]. They also introduce a new lower bound for the rotational case in the same paper. These can be used to deduce new valid inequalities for the upper-level problem. As described in [33], let $\tilde{w}_i^c = \min\{w_i, h_i\}$ if item i can be placed into container c both with and without rotation. Otherwise, $\tilde{w}_i^c = w_i$ or $\tilde{w}_i^c = h_i$ according to the fixed orientation, and define $\tilde{h}_i^c$ analogously. Let a_i denote the total area of item i, i.e. $a_i = w_i \times h_i$.

Continuous lower bound. The so-called continuous lower bound for 2D bin packing is denoted by L_0, and it states that the number of bins of size $W_c \times H_c$ required to place all items $i \in [N]$ of area a_i is at least

$$L_0^c = \left\lceil \frac{\sum_{i\in[N]} a_i}{W_c H_c} \right\rceil \tag{11}$$

By applying this reasoning to the items assigned to each individual container c in the configuration-and-layout problem, where the value of L_0^c must be at most 1, we get that the following constraints must hold for the upper-level problem (1):

$$\frac{\sum_{i\in[N]} \sum_{s\in[S]} u_{c,s,i} a_i}{W_c H_c} \leq 1 \quad \forall c \in [C]. \tag{12}$$

It basically ensures that the total area of the items assigned to container c does not exceed the area of the container.

The L_1 lower bound. Let us define a partitioning of the items for each container $c \in [C]$ as follows, where q is an arbitrary number with $0 \leq q \leq \frac{W_c}{2}$:

$$\begin{aligned} K_1^c(q) &= \left\{ i \in [N] : \tilde{w}_i > W_c - q \right\} \\ K_2^c(q) &= \left\{ i \in [N] : \tfrac{W_c}{2} < \tilde{w}_i \leq W_c - q \right\} \\ K_3^c(q) &= \left\{ i \in [N] : q \leq \tilde{w}_i \leq \tfrac{W_c}{2} \right\} \\ K_4^c(q) &= \left\{ i \in [N] : \tilde{w}_i < q \right\} \end{aligned} \tag{13}$$

Lower bound L_1 on the number of identical bins of size $W_c \times H_c$ required to store all items is as follows:

$$L_1^c = \max_{1\leq q\leq \frac{W_c}{2}} \left\lceil \frac{\sum_{i\in K_1^c(q)\cup K_2^c(q)} \tilde{h}_i^c}{H_c} \right\rceil. \tag{14}$$

Notice that the strongest bound is obtained at some $q \in \left\{ \tilde{w}_i^c : i \in [N] \right\}$, and therefore, it is sufficient to consider those values of q. Similarly to the case of (11) and (12), bound L_1^c can be easily turned into a constraint:

$$\sum_{s\in[S]} \frac{\sum_{i\in K_1^c(q)\cup K_2^c(q)} \tilde{h}_i^c u_{c,s,i}}{H_c} \leq 1 \quad \forall q \in \left\{ \tilde{w}_i^c : i \in [N] \right\} \forall c \in [C]. \tag{15}$$

Observe that interchanging W_c with H_c and simultaneously w_i^c with h_i^c gives similar, valid bounds, and similar constraints can be derived from them.

The L_2 lower bound. With the notation introduced in the previous paragraph, L_2^c lower bound can be defined as

$$L_2^c = \max_{1\leq q\leq \frac{W_c}{2}} \left\lceil \frac{\sum_{i\in K_1^c(q)} \tilde{h}_i^c + \sum_{i\in K_2^c(q)\cup K_3^c(q)} a_i}{H_c} \right\rceil. \tag{16}$$

Again, it is turned into a constraint for the upper-level problem similarly to the L_1^c bounds:

$$\sum_{s\in[S]} \frac{\sum_{i\in K_1^c(q)} \tilde{h}_i^c u_{c,s,i} + \sum_{i\in K_2^c(q)\cup K_3^c(q)} a_i u_{c,s,i}}{H_c} \leq 1 \quad \forall q \in \left\{ \tilde{w}_i^c : i \in [N] \right\} \forall c \in [C]. \tag{17}$$

Once again, another set of constraints can be derived by interchanging W_c with H_c and $\tilde{w}_i^c$ with $\tilde{h}_i^c$.

The L_M^{new} lower bound. Boschetti and Mingozzi [33] introduce the lower bound L_M^{new} that takes into account the possibility of rotations of the items explicitly. Define functions η and μ as follows. Let

$$\eta(s, z, Z) = \begin{cases} \left\lfloor \frac{Z}{s} \right\rfloor - \left\lfloor \frac{Z-z}{s} \right\rfloor, & \text{if } z > \frac{Z}{2} \\ \left\lfloor \frac{z}{s} \right\rfloor, & \text{if } z \leq \frac{Z}{2}, \end{cases} \tag{18}$$

and let

$$\mu_c(j, p, q) = \begin{cases} \min\left\{ \eta(q, w_j, W_c) \times \eta(p, h_j, H_c), \eta(q, h_j, W_c) \times \eta(p, w_j, H_c) \right\} & j \in N_c^* \\ \eta(q, w_j, W_c) \times \eta(p, h_j, H_c) & j \in N_c^0 \\ \eta(q, h_j, W_c) \times \eta(p, w_j, H_c) & j \in N_c^{90}, \end{cases} \tag{19}$$

where N_c^* (respectively, N_c^0 and N_c^{90}) denotes the sets of items that can (resp., cannot and must) be rotated in container c. Lower bound L_M^{new} is defined as

$$L_M^{\text{new}} = \max_{\substack{1\leq p\leq \frac{H_c}{2} \\ 1\leq q\leq \frac{W_c}{2}}} \left\lceil \frac{\sum_{j\in J} \mu(j, p, q)}{\left\lfloor \frac{H_c}{p} \right\rfloor \left\lfloor \frac{W_c}{q} \right\rfloor} \right\rceil. \tag{20}$$

Accordingly, the following constraint can be derived for the upper-level problem:

$$\sum_{s\in[S]} \frac{\sum_{i\in[N]} \mu(i, p, q) u_{c,s,i}}{\left\lfloor \frac{H_c}{p} \right\rfloor \left\lfloor \frac{W_c}{q} \right\rfloor} \leq 1 \quad \forall c \in [C], \tag{21}$$

for all $p \in \left\{ h_i : h_i \leq \frac{H_c}{2} \right\}$ and $q \in \left\{ w_i : w_i \leq \frac{W_c}{2} \right\}$.

6. Computational experiments

6.1. Design of experiments

In order to evaluate the computational efficiency of the proposed techniques, four variants of the proposed Benders approach were implemented and tested on the FMS design problem. Each variant is an extension of the previous ones, and the proposed computational techniques are added to the Benders framework in an increasing order of development effort:

Benders: The first and simplest variant solves the Benders master problem extended with constraints derived from the trivial L_0 lower bound as in (12), identifies the cells that admit no feasible layout, and then extends the upper-level problem with no-good cuts of form (4).

Benders+SB: The second variant improves the baseline approach by extending the upper-level problem with *symmetry breaking* as described in Section 5.2.

Benders+SB+LC: The third variant strengthens the Benders approach by extending the upper-level problem with *lifted cuts* of the form (7) upon infeasibility in the layout sub-problem.

Benders+SB+LC+RC: The fourth, complete variant improves the formulation of the upper-level problem further by extending it with all *redundant constraints* as described in Section 5.3 (including mutually exclusive item pairs).

Moreover, as an alternative solution approach, a monolithic MILP formulation of the FMS design problem was composed by merging the configuration and the layout sub-problems into a single, huge MILP model. This required adding new constraints and auxiliary binary variables to establish the connection between the two sub-problems.

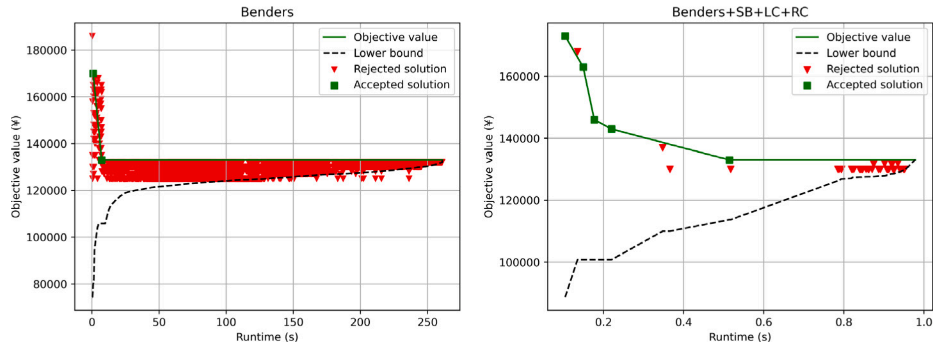

Fig. 5. Evolution of the solutions over time in case of the basic (left) and the most advanced (right) Benders approaches. Observe the different time scales on the horizontal axis.

The monolithic models were solved using the default branch-and-bound algorithm of a commercial MILP solver. Observe that a part of the proposed computational techniques developed originally for the Benders approach, namely, symmetry breaking and redundant constraints, can be applied trivially to the monolithic MILP model as well. On the contrary, lifted cuts are not applicable. Accordingly, the following two variants of monolithic MILP were included in the experiments:

Monolithic: The first monolithic MILP model includes the merged configuration and layout sub-problems, as well as the constraints derived from the trivial L_0 lower bound (12).

Monolithic+SB+RC: The second, advanced monolithic MILP formulation includes all techniques applicable to this approach, i.e., *symmetry breaking* (Section 5.2) and all *redundant constraints* (Section 5.3).

The algorithms were implemented in Python, using FICO Xpress[1] v9.2.5 as a MILP solver. The experiments were performed on a server with i9-7960X CPU @ 2.80 GHz and Linux operating system using one thread, with a limit on the total computation time of 7200 s for each problem instance, for every variant of the solution method.

6.2. Problem instances

In order to perform the computational experiments on a sufficiently large set of problem instances of different dimensions, which are industrially relevant at the same time, random instances were generated based on data from the real industrial case study presented in Section 7, involving the machining of a family of cylindrical parts. The original data set contained detailed characteristics of 30 products (dimensions), as well as 6 machine types and 6 robot types (capabilities, speed, dimensions, and costs). Based on the original data, random instances of different sizes were generated. In particular, the number of products, P, was chosen from $\{5, 30\}$; the number of cells, C, from $\{3, 4, 6\}$; and finally, the number of slots per cell, S, from $\{3, 4, 5\}$, including one robot slot and $S - 1$ machine slots. For all problem sizes, 10 instances were generated, resulting in 180 instances altogether. Process and handling times were set by applying a random perturbation to the real industrial values. In order to ensure that a feasible solution exists, first, a system configuration, also including the assignment of products to machines was randomly created, and then demands were set to ensure that the given random configuration produces the desired mix of products by the end of the monthly time horizon. Finally, cell dimensions were determined in such a way that the selected items can be located in a multi-row arrangement. It is highlighted that the random configuration assumed during generation is a feasible, but not necessary optimal solution of the system configuration problem, since a different configuration of lower cost might also be able to produce the desired mix of products.

[1] https://www.fico.com/en/products/fico-xpress-optimization

6.3. Illustration on a sample instance

First, the Benders approach is illustrated on the small sample instance whose optimal solution was presented earlier in Fig. 2. The instance contains 3 products, at most 4 cells (only 3 cells must be built in the optimal solution), 4 types of machines and robots, as well as 4 machine slots and one robot slot in each cell (at most 2 machine slots and exactly one robot slot are actually used per cell). Two variants of the solution method, the baseline *Benders* and the most advanced *Benders+SB+LC+RC* were evaluated on the instance, with a time limit of 7200 s. Fig. 5 displays the progress of the branch-and-cut search procedure over time for the two variants. The upper, green curve represents the objective value of the best solution found, whereas the lower, dashed curve shows the lower bound. The meeting point of the two curves indicates that the optimality of the solution is proven, and search can terminate. According to the Benders approach, the search for the optimal upper-level solution discovers from time to time candidate improving configurations, which are then submitted to the lower-level layout planner for evaluation. Candidate configurations rejected due to layout infeasibility are indicated by red triangles, whereas those accepted with a feasible layout are shown by green squares. The comparison of the two diagrams reveals the efficiency of the developed inference techniques: the baseline *Benders* variant required 228.63 s and 41,263 candidate configurations to arrive at a proven optimal solution, while *Benders+SB+LC+RC* achieved the same in only 1.26 s with 54 candidate configurations.

A detailed investigation of the performance of the two algorithms is presented in Table 3. The results show that, although both variants solved this small sample instance to proven optimality, the baseline *Benders* variant needed two orders of magnitude more computation time to achieve this than the most advanced variant (228.63 s versus 1.26 s). Most of the search effort was spent on the upper-level configuration problem, while the lower-level problem took 18.77 s, including cache search and administration of rejected and accepted solutions (17.63 s) and solving the layout planning MIP (1.14 s). Accordingly, the developed inference techniques could decrease the size of the branch-and-cut search tree by three orders of magnitude, in terms of the number of search nodes, simplex iterations, cuts, and the number of candidate solutions, too. It is noticeable that the inference techniques decreased primarily the number of rejected configurations, since they address identifying and eliminating possible configurations that do not admit a feasible layout. In contrast, the number of accepted configurations is not affected directly: it may increase or decrease slightly due to the interference of the added inference techniques with the search strategy of the commercial solver.

6.4. Computational results

The results over the set of 180 randomly generated problem instances are displayed in Table 4 for the four Benders variants and in Table 5 for the two monolithic MILP variants. Each row of the

Table 3
Summary of results on the sample instance.

	Time (s)		B&C tree search			Candidate solutions	
	Total	Lower	Nodes	Simplex it.	Cuts	Rejected	Accepted
Benders	228.63	18.78	232,563	3,114,691	41,277	41,261	2
Benders+SB+LC+RC	1.26	0.37	225	9,759	108	49	5

Table 4
Computational results with the four variants of the Benders approach.

			Benders			Benders+SB			Benders+SB+LC			Benders+SB+LC+RC		
P	C	S	Time (s)	Gap	Opt	Time (s)	Gap	Opt	Time (s)	Gap	Opt	Time (s)	Gap	Opt
5	3	3	13	0.00%	10	6	0.00%	10	5	0.00%	10	1	0.00%	10
		4	250	0.00%	10	18	0.00%	10	16	0.00%	10	5	0.00%	10
		5	4,945	2.22%	4	161	0.00%	10	149	0.00%	10	57	0.00%	10
	4	3	298	0.00%	10	41	0.00%	10	43	0.00%	10	3	0.00%	10
		4	5,375	2.05%	3	383	0.00%	10	398	0.00%	10	88	0.00%	10
		5	7,200	6.71%	0	3,862	0.49%	8	3,639	0.41%	9	1,346	0.36%	9
	6	3	5,979	3.00%	3	2,731	0.61%	8	2,709	0.33%	8	51	0.00%	10
		4	7,200	6.78%	0	6,311	2.21%	4	6,397	2.05%	4	2,710	0.51%	8
		5	7,200	9.58%	0	7,200	5.76%	0	7,200	6.04%	0	6,149	3.07%	2
30	3	3	18	0.00%	10	11	0.00%	10	11	0.00%	10	3	0.00%	10
		4	818	0.00%	10	132	0.00%	10	128	0.00%	10	21	0.00%	10
		5	5,422	1.99%	4	517	0.00%	10	518	0.00%	10	190	0.00%	10
	4	3	631	0.00%	10	329	0.00%	10	367	0.00%	10	6	0.00%	10
		4	7,200	5.68%	0	3,191	0.46%	8	3,173	0.52%	8	457	0.00%	10
		5	7,200	9.79%	0	7,205	4.46%	0	7,200	4.57%	0	5,812	1.25%	5
	6	3	6,682	7.17%	1	5,755	2.56%	4	6,072	2.42%	4	446	0.00%	10
		4	7,200	13.64%	0	7,200	9.92%	0	7,200	10.00%	0	7,200	7.34%	0
		5	7,200	17.19%	0	7,200	11.94%	0	7,200	11.92%	0	7,200	9.36%	0
Total			4,491	4.77%	75	2,903	2.13%	122	2,912	2.13%	123	1,764	1.22%	144

Table 5
Computational results with the two variants of the monolithic MILP approach.

			Monolithic			Monolithic+SB+RC		
P	C	S	Time (s)	Gap	Opt	Time (s)	Gap	Opt
5	3	3	3	0.00%	10	1	0.00%	10
		4	153	0.00%	10	11	0.00%	10
		5	4,097	1.37%	6	223	0.00%	10
	4	3	24	0.00%	10	6	0.00%	10
		4	3,614	1.29%	7	330	0.00%	10
		5	7,200	6.35%	0	4,322	0.82%	8
	6	3	1,192	0.50%	9	198	0.00%	10
		4	7,200	8.37%	0	5,662	2.08%	4
		5	7,200	12.99%	0	7,200	6.23%	0
30	3	3	5	0.00%	10	2	0.00%	10
		4	156	0.00%	10	30	0.00%	10
		5	3,995	0.47%	9	300	0.00%	10
	4	3	28	0.00%	10	6	0.00%	10
		4	6,550	1.98%	4	601	0.00%	10
		5	7,200	10.20%	0	6,090	1.33%	4
	6	3	2,327	0.30%	9	509	0.00%	10
		4	7,200	14.11%	0	7,200	6.94%	0
		5	7,200	18.49%	0	7,200	9.82%	0
Total			3,630	4.24%	104	2,216	1.51%	136

tables corresponds to a given problem size, and each group of columns contains the results for one variant of the solver. In the *Problem size* columns, P, C, and S denote the number of products, cells, and slots per cell, respectively. Columns *Time* display the average computation time in seconds, whereas columns *Gap* show the average optimality gap. Averages are computed over all instances, including those solved to optimality and those where the time limit was hit. Finally, columns *Opt* contain the number of problem instances solved to proven optimality out of 10 for the given problem size. The results are also visualized in Fig. 6, where separate diagrams stand for the three performance measures of computation time, optimality gap, and number of instances solved to optimality, as well as for instance sizes $P = 5$ and $P = 30$.

While the smallest instances could be solved to optimality by using any of the variants (10 out of 10 instances solved), the largest instances illustrate the limits of the proposed approach (no optimal solutions found even by the most advanced variant). Yet, even for these challenging instances, the solvers found reasonable solutions that are feasible according to both the upper-level and the lower-level constraints. The

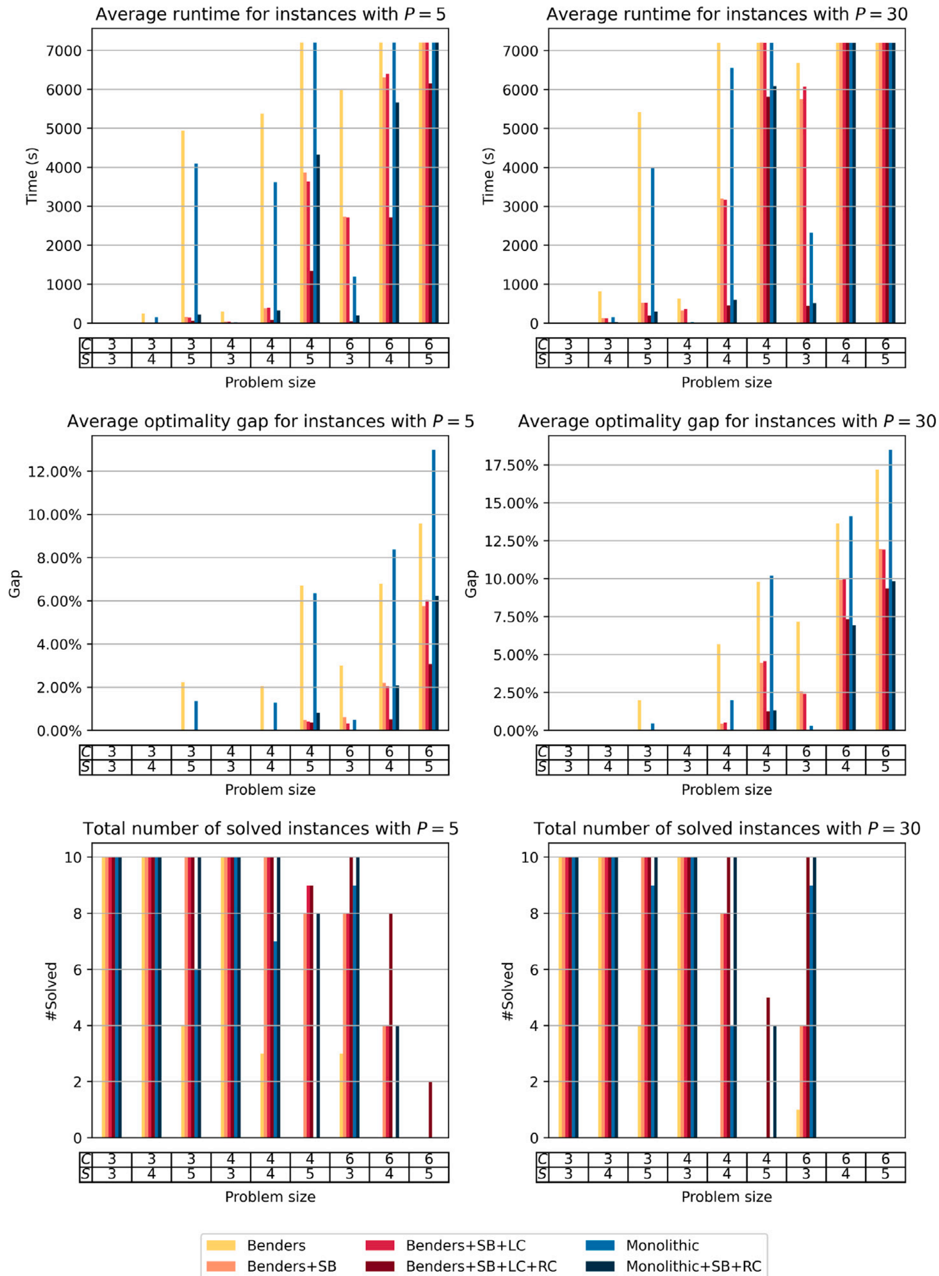

Fig. 6. Summary of computational results. The horizontal axis is divided by problem sizes, with the same notation as Tables 4 and 5.

difficulty of an instance depends clearly on the number of cells and the number of slots per cell, but at the same time, less on the number of products.

Remarkably, for at most four cells ($C \leq 4$) and four slots per cell ($S \leq 4$), the most advanced variant could solve all instances to proven optimality, and this took only 73 s of computation time on average.

The results show that each presented computational technique could improve the performance of the Benders approach. Notably, adding symmetry breaking increased the total number of optimal solutions from 75 to 122, decreased the average gap from 4.77% to 2.13%, and decreased the average computation time from 4,491 s to 2,903 s. Redundant constraints increased the number of optimal solutions further from 123 to 144, decreased the gap from 2.13% to 1.22%, and decreased time from 2,912 s to 1,764 s. In this setting, the impact of lifted cuts was marginal: they increased the number of optimal solutions from 122 to 123. Still, it must be mentioned that in a somewhat different (and overall, less efficient) earlier implementation of the Benders approach that restarted the upper-level search each time a cut was added, lifted cuts also had a huge impact. The explanation is that in the earlier implementation, stronger cuts affected the repeated traversals of the entire search tree, whereas with the current implementation, the impact occurs only on a smaller portion of the search tree branches.

The alternative, monolithic MILP approach performed surprisingly well: the simplest *Benders* variant was even outperformed by the baseline *Monolithic* variant (75 versus 104 optimal solutions). Adding the developed inference techniques, namely, symmetry breaking and redundant constraints also boosted the performance of the monolithic approach (from 104 to 136 optimal solutions). Nevertheless, when comparing the most advanced monolithic and Benders variants, *Monolithic+SB+RC* and *Benders+SB+LC+RC*, it turns out the latter dominates according to all performance measures: 136 versus 144 optimal solutions, 1.51% versus 1.22% average gap, and 2,216 s versus

1,764 s average computation time. This comparison also demonstrates that strong inference techniques can have a great impact on search performance almost independently of the solution approach.

6.5. Discussion on alternative solution approaches

While the authors are not aware of earlier contributions in the literature that study the same integrated configuration-and-layout problem, various alternative approaches might be applicable for solving it. Accordingly, two such approaches, namely *genetic algorithms* (GA) and *mixed-integer nonlinear programming* (MINLP) were also evaluated on the above FMS design problem instances.

The GA solution approach was implemented in the widely used genetic algorithm toolkit PyGAD[2]. The natural genetic encoding was used, with each gene corresponding to a variable of the monolithic MILP, and the default genetic operators were applied. Various combinations of the starting parameters were examined, including population size, number of mating parents, elitism, as well as parameters for the adaptive mutation rate. Two versions of the GA were experimented: a single-objective GA where the penalty for constraint violation and the original objective function were combined into a single criterion, and a multi-objective GA that handles these two criteria separately. Despite the thorough experimentation, the GAs did not manage to find a feasible solution even on the smallest instances within the prescribed time limit of 7200 s.

Likewise, the FMS design problem was formulated as a MINLP in a straightforward manner. The upper-level MILP was extended with variables describing the layout of the selected items, i.e., binary variables for the pairwise relative position and the rotation of the items, and continuous variables for their absolute positions. Non-linearities arise when layout feasibility is expressed via constraints involving the product of two or more binary variables. The resulting formulation is smaller in the number of variables and constraints than the monolithic MILP. This MINLP formulation was submitted to the state-of-the-art commercial MINLP solver called Hexaly,[3] which is claimed to be the best off-the-shelf solver for routing, scheduling, and packing problems, and uses exact algorithms as well as heuristics in a hybrid framework. Despite this, the solver did not find optimal solutions, and in many cases even a feasible solution, for the problem instances within the 7200 s time limit.

For the above reasons, the detailed computational results achieved using these alternative approaches are omitted from this paper. While it is possible that, with the addition of custom algorithms (e.g., problem-specific genetic encoding and operators for the GA), these approaches could be developed into efficient solvers, this would require extensive research on its own right, and hence, these are out of the scope of the present paper.

7. Industrial case study

The proposed methods were developed in an industry-academia collaborative project focusing on decision support for production system design. The methods were evaluated on data from a real customer project from the past involving the design of an FMS for machining a family of cylindrical products. The actual model implemented in the industrial application fits into the generic configuration-and-layout model introduced in Section 3, yet, in order to capture all relevant practical requirements, it is significantly richer than the specific FMS design application described in Section 4. Due to the numerous but mathematically straightforward extensions, for the sake of brevity and confidentiality, the precise mathematical model for those extensions is omitted in this paper. A brief textual description is given below.

The main difference is the application of a significantly more elaborate model of the resources required in the machining cells. In addition to the robots (5 types) and CNC machine tools (5 types) also considered above, items also included part stockers (3 types), gripper hand stockers (3 types), and fixture stockers (3 types), temporal storages (3 types), as well as dedicated gripper hands for each product (5 types for 5 products, captured only in the configuration sub-problem, while omitted in the layout sub-problem). The model was also extended with various constraints on the compatibility of those items, e.g., gripper hand stockers are required only in cells where multiple types of products are manufactured; material handling times in constraint (2h) are increased with an estimation of changeover times in such cells. The model assumes imperfect utilization rates of the robots and machines to account for blocking situations and maintenance. The layout planning sub-problem was also extended with two additional constraints: all relevant items must be located within the reach of the robot according to the maximum metric (also called the L_∞ norm), and the robot must be located on the front side of the machine to access the machine workspace through the door. Since all other items are low, the robot can approach them from above. Accordingly, the partial ordering of robot items in the lifted cuts had to take into account robot reach, the partial ordering of machines had to differentiate the front from the side, while an unchanged ordering could be applied to all other items. Symmetry breaking was implemented with interchangeable slots corresponding to slots dedicated to the same type of items. The solver used in the industrial case study was based on an earlier version of the methods presented in the paper, with an implementation of the continuous lower bound (12), but without the stronger bounds from Section 5.3, corresponding to variant *Base+SB+LC* above. Moreover, the Benders approach was implemented by restarting the upper-level solver each time a cut was generated, instead of the more advanced branch-and-cut algorithm described in this paper.

In the case study, the total production volume for the 5 different products was 90 000 pieces per year. There was room for 10 new rectangular cells of different dimensions on the shop floor, though, building 6 new cells was sufficient to satisfy customer demand. The optimal configuration for this case study was found in ca. 23 min in 154 Benders iterations. The configurations and layouts computed for the 6 cells are displayed in Fig. 7. From this plan, a 3D simulation model of the manufacturing system was constructed and evaluated by the engineers of the company, see Fig. 8. Their assessment confirmed the correctness of the computed results and the overall validity of the approach.

An important aspect of the evaluation was the efficiency of the decision support provided to the human designer. For this purpose, a prospective user of the developed system, a junior designer with 3 months of experience, was asked to solve some actual FMS design problems by using the developed prototype. The problems arrived from customers, and they were similar to (but typically smaller in size than) the case study presented above. By manual planning according to the current practice, the estimated working time for solving these problems was 28 h on average. With the support of the prototype, this was reduced to 2.2 h on average, which included data preparation and actual computations, as well as visualizing the results in simulation software. Obviously, these values are highly dependent on the task at hand and the particular engineer.

8. Conclusions and future research

8.1. Conclusions of the current study

This paper introduced a novel exact method for integrated configuration-and-layout problems. While the specific application and the industrial case study focused on the configuration of flexible manufacturing systems, the generic method can find applications in various fields that require jointly optimizing the selection and geometrical

[2] https://pygad.readthedocs.io/en/latest/
[3] https://www.hexaly.com/

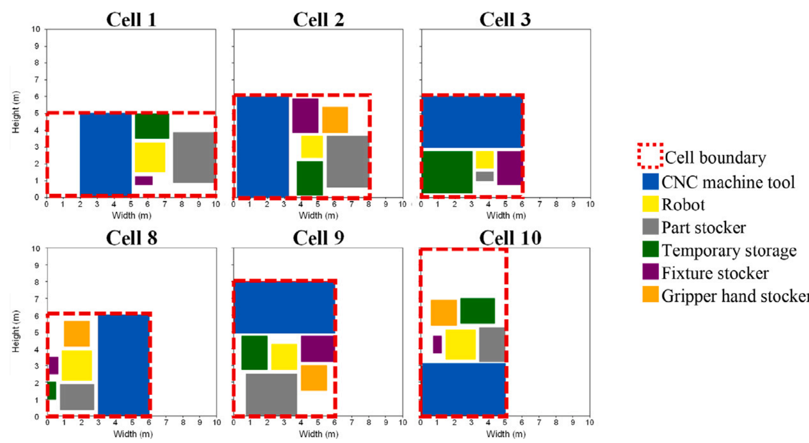

Fig. 7. Configurations and 2D layouts for the 6 new cells in the industrial case study.

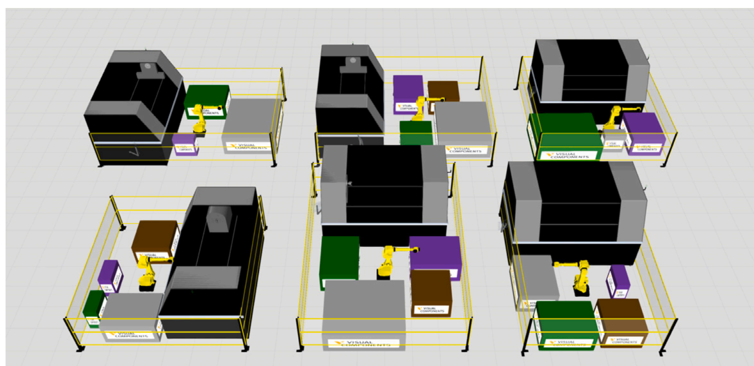

Fig. 8. 3D simulation model of the designed FMS cells. CNC machines in black, part stockers in gray, fixture stockers in purple, temporal storages in green, hand stockers in brown. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

arrangement of items. The integrated problem is a computationally challenging combinatorial optimization problem, and finding optimal solutions requires well-designed, efficient solution algorithms. For this purpose, various inference techniques, including symmetry breaking, lifted cuts, as well as redundant constraints inspired by known lower bounds were added to the baseline logic-based Benders solution method. Computational experiments demonstrated that the resulting algorithm finds proven optimal solutions for problem sizes relevant in manufacturing applications. In the real industrial case study, the approach computed high-quality configurations and layouts, also validated by human experts, while it substantially decreased the engineering work hours required for manufacturing system design.

8.2. Extensions of the model

There are numerous straightforward extensions of the presented mathematical model. These include more precise, rectilinear geometries for containers (manufacturing cells) and items (robots and machines). All the presented inference techniques remain valid with this extension, though, some implementation details must be adjusted, including the definition of the partial orderings in the lifted cuts and the computation of the lower bounds. Moreover, an optimization criterion can be added to the layout sub-problem as a secondary criterion in the overall configuration-and-layout problem.

Another relevant direction is capturing richer practical requirements on the layout via the introduction of so-called *layout templates* that prescribe constraints on the absolute and relative positions of the items placed in the given slots. However, the adaptation of symmetry breaking (both for absolute and relative position constraints) and lifted cuts (in case of absolute position constraints) is not straightforward.

8.3. Extensions of the solution approach

In addition to various extensions of the model, the further enhancement of the algorithms is also an intriguing direction. The identification of *minimal conflict sets* for further lifted cuts was found to be an efficient technique for similar problems [34,35], yet, with simpler upper-level problems and significantly higher number of items per container. In our experiments with a low number of items, this technique did not show any improvement over the variants presented in the paper.

A direction for future research, particularly relevant for practical applications, is the development of algorithms for solving large-size problems. This can involve both meta-heuristic approaches for the upper-level configuration sub-problem within the Benders framework, also making use of the proposed techniques to decrease the search effort; as well as repair heuristic that, upon infeasibility in the lower-level layout sub-problem, modify the configuration to ensure layout-feasibility. Although both techniques may result in losing the exactness of the overall solution approach, this is a reasonable price for achieving feasible solutions in a computationally efficient way.

CRediT authorship contribution statement

Péter Dobrovoczki: Writing – review & editing, Writing – original draft, Visualization, Software, Investigation, Formal analysis, Conceptualization. **András Kovács:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Investigation, Formal analysis, Conceptualization. **Hiroyuki Sakata:** Validation, Resources, Data curation, Conceptualization. **Daisuke Tsutsumi:** Validation, Resources, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors are grateful to Zsombor Szádoczki and to the anonymous reviewers for their comments on an earlier version of this manuscript. Hungarian authors were supported by the 2021-17376/NP/BILAT_HU_DE_1 and the TKP2021-NKTA-01 grants.

References

[1] Dolgui A, Proth J-M. Supply chain engineering: Useful methods and techniques. Springer; 2010.
[2] Hooker JN, Ottosson G. Logic-based Benders decomposition. Math Program A 2003;96(1):33–60.
[3] Weckenborg C, Schumacher P, Thies C, Spengler TS. Flexibility in manufacturing system design: A review of recent approaches from operations research. European J Oper Res 2024 315(2):413–41.
[4] Matta A, Tolio T, Karaesmen F, Dallery Y. An integrated approach for the configuration of automated manufacturing systems. Robot Comput-Integr Manuf 2001;17(1):19–26.
[5] Monitto M, Pappalardo P, Tolio T. A new fuzzy AHP method for the evaluation of automated manufacturing systems. CIRP Ann 2002;51(1):395–8.
[6] Sivalingam C, Subramaniam SK. Cobot selection using hybrid AHP-TOPSIS based multi-criteria decision making technique for fuel filter assembly process. Heliyon 2024;10(4):e26374.
[7] Battaïa O, Dolgui A, Guschinsky N. Optimal cost design of flow lines with reconfigurable machines for batch production. Int J Prod Res 2020;58(10):2937–52.
[8] Abbas M, ElMaraghy H. Synthesis and optimization of manufacturing systems configuration using co-platforming. CIRP J Manuf Sci Technol 2018;20:51–65.
[9] Li S, Wang H, Hu SJ, Lin Y-T, Abell JA. Automatic generation of assembly system configuration with equipment selection for automotive battery manufacturing. J Manuf Syst 2011;30(4):188–95.
[10] Becker C, Scholl A. A survey on problems and methods in generalized assembly line balancing. European J Oper Res 2006;168(3):694–715.
[11] Boysen N, Fliedner M, Scholl A. A classification of assembly line balancing problems. European J Oper Res 2007;183(2):674–93.
[12] Boysen N, Schulze P, Scholl A. Assembly line balancing: What happened in the last fifteen years? European J Oper Res 2022;301(3):797–814.
[13] Battaïa O, Dolgui A. A taxonomy of line balancing problems and their solution approaches. Int J Prod Econ 2013;142(2):259–77.
[14] Scholl A, Becker C. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. European J Oper Res 2006;168(3):666–93.
[15] Akpinar S, Elmi A, Bektaş T. Combinatorial Benders cuts for assembly line balancing problems with setups. European J Oper Res 2017;259(2):527–37.
[16] Sikora CGS. Benders' decomposition for the balancing of assembly lines with stochastic demand. European J Oper Res 2021;292(1):108–24.
[17] Şahin M, Kellegöz T. Benders' decomposition based exact solution method for multi-manned assembly line balancing problem with walking workers. Ann Oper Res 2023;321(1):507–40.
[18] Michels AS, Lopes TC, Sikora CGS, Magatão L. A Benders' decomposition algorithm with combinatorial cuts for the multi-manned assembly line balancing problem. European J Oper Res 2019;278(3):796–808.
[19] Tsutsumi D, Kovács A, Szalóki Á. Novel heuristic approach to integrating task sequencing and production system configuration. Procedia CIRP 2022;107:28–33.
[20] Battaïa O, Dolgui A. Hybridizations in line balancing problems: A comprehensive review on new trends and formulations. Int J Prod Econ 2022;250:108673.
[21] Albus M, Huber MF. Resource reconfiguration and optimization in brownfield constrained robotic assembly line balancing problems. J Manuf Syst 2023;67:132–42.
[22] Selim HM, Askin RG, Vakharia AJ. Cell formation in group technology: Review, evaluation and directions for future research. Comput Ind Eng 1998;34(1):3–20.
[23] Mahdavi I, Teymourian E, Baher NT, Kayvanfar V. An integrated model for solving cell formation and cell layout problem simultaneously considering new situations. J Manuf Syst 2013;32(4):655–63.
[24] Chiang C-P, Lee S-D. Joint determination of machine cells and linear intercell layout. Comput Oper Res 2004;31(10):1603–19.
[25] Chang C-C, Wu T-H, Wu C-W. An efficient approach to determine cell formation, cell layout and intracellular machine sequence in cellular manufacturing systems. Comput Ind Eng 2013;66(2):438–50.
[26] Mohammadi M, Forghani K. A novel approach for considering layout problem in cellular manufacturing systems with alternative processing routings and subcontracting approach. Appl Math Model 2014;38(14):3624–40.
[27] Ji D, Zhang Z, Liang W, Wang C, He Z. Mathematical formulation and a novel two-stage algorithm for double-row layout problem with fixed loading and unloading points. J Manuf Syst 2023;69:242–54.
[28] Pablo Pérez-Gosende JM, Díaz-Madroñero M. Facility layout planning. An extended literature review. Int J Prod Res 2021;59(12):3777–816.
[29] Doan NCN, Lin W. Optimal robot placement with consideration of redundancy problem for wrist-partitioned 6R articulated robots. Robot Comput-Integr Manuf 2017;48:233–42.
[30] Ghungrad S, Mohammed A, Haghighi A. Energy-efficient and quality-aware part placement in robotic additive manufacturing. J Manuf Syst 2023;68:644–50.
[31] Iori M, de Lima VL, Martello S, Miyazawa FK, Monaci M. Exact solution techniques for two-dimensional cutting and packing. European J Oper Res 2021;289(2):399–415.
[32] Lodi A, Martello S, Monaci M. Two-dimensional packing problems: A survey. European J Oper Res 2002;141(2):241–52.
[33] Boschetti MA, Mingozzi A. The two-dimensional finite bin packing problem. Part II: New lower and upper bounds. Q J Belg Fr Ital Oper Res Soc 2003;1(2):135–47.
[34] Côté J-F, Dell'Amico M, Iori M. Combinatorial Benders' cuts for the strip packing problem. Oper Res 2014;62(3):643–61.
[35] Delorme M, Iori M, Martello S. Logic based Benders' decomposition for orthogonal stock cutting problems. Comput Oper Res 2017;78:290–8.
[36] Iori M, Martello S. Routing problems with loading constraints. Trans Oper Res 2010;18(1):4–27.
[37] Iori M, González JJS, Vigo D. An exact approach for the vehicle routing problem with two-dimensional loading constraints. Transp Sci 2007;41(2):253–64.
[38] Melega GM, de Araujo SA, Jans R. Classification and literature review of integrated lot-sizing and cutting stock problems. European J Oper Res 2018;271(1):1–19.
[39] Gramani M, França P, Arenales M. A Lagrangian relaxation approach to a coupled lot-sizing and cutting stock problem. Int J Prod Econ 2009;119(2):219–27.
[40] Gramani MCN, França PM, Arenales MN. A linear optimization approach to the combined production planning model. J Franklin Inst 2011;348(7):1523–36.
[41] Vanzela M, Melega GM, Rangel S, de Araujo SA. The integrated lot sizing and cutting stock problem with saw cycle constraints applied to furniture production. Comput Oper Res 2017;79:148–60.
[42] Gramani MCN, França PM. The combined cutting stock and lot-sizing problem in industrial processes. European J Oper Res 2006;174(1):509–21.
[43] Gilmore PC, Gomory RE. Multistage cutting stock problems of two and more dimensions. Oper Res 1965;13(1):94–120.
[44] Boschetti MA, Mingozzi A. The two-dimensional finite bin packing problem. Part I: New lower bounds for the oriented case. Q J Belg Fr Ital Oper Res Soc 2003;1(1):27–42.