An adversarial approach for the mixed-model assembly line design with new product variants in production generations

S. Ehsan Hashemi-Petroodi, Yosra Mezghani, Simon Thevenin, Alexandre Dolgui

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 58-19 (2024) 97–102

An adversarial approach for the mixed-model assembly line design with new product variants in production generations ⋆

S. Ehsan Hashemi-Petroodi * **Yosra Mezghani** **
Simon Thevenin ** **Alexandre Dolgui** **

* *KEDGE Business School Campus Bordeaux, 33405 Talence, France (e-mail: seyyed-ehsan.hashemi-petroodi@kedgebs.com).*
** *IMT Atlantique, LS2N - CNRS, La Chantrerie, 4, rue Alfred Kastler - B.P. 20722, F-44307 Nantes Cedex 3, France (e-mail: yosra.mezghani@etudiant-enit.utm.tn - simon.thevenin@imt-atlantique.fr - alexandre.dolgui@imt-atlantique.fr).*

Abstract: Assembly lines typically operate for several decades. Process engineers reconfigure the lines several times during the product family's life cycle, whereas product families may change several times a year in response to sales and marketing demands. These reconfigurations are often expensive and inefficient if the line is not flexible enough. The current study explores the feasibility of creating a line that takes product evolution into account during the line's life cycle. We study a line where a worker/robot and equipment pieces required are located at each station. When a new product model replaces one of the current variants in the product family, the line reconfigures to produce different product models from the same family. Reconfiguration can re-assign some tasks and rearrange equipment and resource elements. We formulate a model that accounts for the uncertainty of the product family evolution and the market demand. We propose an adversarial approach for the robust optimization of the mixed-model assembly line design for the worst-case scenario from a scenario tree for the future product family requirements. We run computational experiments using benchmark data. The results demonstrate that the developed adversarial approach outperforms the classical methods from the literature in terms of CPU time and solution quality.

Keywords: Robust optimization, Mixed model assembly line, Reconfigurability, Product family evolution, Adversarial approach

1. INTRODUCTION

In today's manufacturing, companies are facing increasing requests for product customization and the rapid fluctuations of market demand. Consequently, assembly lines undergo frequent adaptations to accommodate changes in customer preferences, product specifications, and technological advancements. Therefore, companies are willing to design assembly lines that facilitate the seamless integration of new technologies or resources and are easily reconfigurable to meet future market changes and product family evolution. This study focuses on Reconfigurable Manufacturing Systems (RMS) (Koren et al., 1999). While RMS offers flexibility, designing systems capable of accommodating all product variants within a family remains a challenge. This study proposes a cost-effective robust optimization approach to facilitate the design and reconfiguration planning of assembly lines with evolving products.

Aligned with our European-funded project on developing AI-based software for agile manufacturing environments, this study focuses on the complexities of anticipating and mitigating the impact of product variant changes on assembly line design. Drawing insights from industrial partners, particularly in the automotive sector, the research is built on addressing the challenges posed by the uncertainty that arises from the future evolution of the product family. In this study, we propose robust optimization methods that design lines that are robust to the uncertainty in the evolution of product models during the life cycle of assembly lines. The problem aims at minimizing total design and reconfiguration costs for the worst-case scenario. Hashemi-Petroodi et al. (2022) proposed a mixed integer linear programming (MILP), seeking to optimize resource allocation, equipment installation, and workforce management for the worst scenario of product family evolution considering a created sample-based scenario tree. In the current study, we compare the results of the MILP presented in Hashemi-Petroodi et al. (2022) with the adversarial approach developed. This later approach enhances solution

⋆ The present work was conducted within the project ASSISTANT (https://assistant-project.eu/) that is funded by the European Commission, under grant agreement number 101000165,H2020–ICT -38-2020, artificial intelligence for manufacturing.
Corresponding author: S. Ehsan Hashemi-Petroodi (seyyed-ehsan.hashemi-petroodi@kedgebs.com)

10.1016/j.ifacol.2024.09.101

robustness, offering faster insights into resilient system configurations, compared to the developed MILP. Moreover, computational experiments and results illustrate the superiority of such a robust approach in mitigating design and reconfiguration costs compared to the classical models in which reconfiguration planning decisions are made at each period.

2. LITERATURE REVIEW

The literature review classifies assembly lines into single-, mixed-model, and multi-model categories, each catering to different production scenarios. While single lines streamline the production of a single product type, mixed-model lines offer more flexibility by accommodating multiple product variants with minimal setup times between models (Sivasankaran and Shahabudeen, 2017). Mixed-model lines can benefit from reconfigurability existing in RMSs introduced by (Koren et al., 1999). The concept of RMS relies on reconfigurable tools that allow the rapid adaptation of the line in the face of changing production requirements by re-arranging production components, efficiently and less costly. However, optimization of the assembly line design and future reconfiguration planning in the face of uncertainty remains a challenge (Manzini et al., 2018). We study the evolving nature of assembly line design and reconfiguration planning, particularly considering the uncertainty of product family evolution.

Uncertainties in assembly lines present significant and real challenges to manufacturers, necessitating innovative approaches to mitigate their impact. The literature demonstrates various sources of uncertainty, ranging from fluctuations in product demand to unpredictable changes in product variants and technological requirements (Zhang et al., 2023). These uncertainties complicate assembly line design and balancing. Notably, there are only a few studies addressing uncertainties in assembly line reconfiguration, particularly concerning product family evolution (Wei et al., 2017; Biswas et al., 2023).

In response to the complexities of assembly line design with uncertainties, we propose an adversarial approach commonly used in robust optimization principles. Robust optimization optimizes assembly line performance for the worst-case scenarios of uncertainty (Liu et al., 2020). Recently, Hashemi-Petroodi et al. (2022) developed a mathematical model for the robust optimization of the design and reconfiguration planning of a mixed-model line with the uncertainty of the future product family evolution. However, our current work significantly contributes compared to this previous study. We develop an adversarial approach with corresponding computational tests to solve the problem for larger instances more efficiently, as well as more computational experiments and insights are provided in this work compared to Hashemi-Petroodi et al. (2022). Moreover, we provide a computational comparison of the proposed robust model with the classical approach that designs and reconfigures the line period-by-period by accounting only for the current product family.

3. PROBLEM DESCRIPTION

This section describes the considered mixed-model assembly line reconfiguration planning problem ($MALRP$). The problem comes from a real-world challenge encountered by an automotive producer in France, who periodically reconfigures its mixed-model line approximately every 6 to 12 months. The reconfiguration happens often during weekends or public holidays due to the costs and complexities involved. This reconfiguration aims to accommodate market changes, introduce new product variants, and adjust to shifts in demand for existing product models within the product family. Such reconfigurations belong to the relocation, addition, or removal of resources and equipment at various stations along the assembly line, reflecting the company's endeavor to optimize the design and reconfiguration planning while considering long-term investment costs and amortization.

In each generation g in the set of generation $\mathcal{G} = \{0 \dots G\}$, the line may assemble a product family from a set $\mathcal{PS}_g = \{1 \dots PS_g\}$ of possible product families. Note that $g = 0$ refers to the current generation and the set of product models PS_0 is known ($|PS_0| = 1$). Each product family p at generation g contains the set $\mathcal{I}_{pg} = \{1 \dots I_{pg}\}$ of product models. We denote by m^g_{ip}, the market demand volume of the existing product model i in product family p at generation g. On the contrary, each future generation includes different scenarios that represent different evolutions of the product families. For each product family p, we create a joint precedence graph A_p with a weighted process time of tasks considering the demand of each product model in the family, as explained in Bryan et al. (2013); Hashemi-Petroodi et al. (2022). The graph A_p includes precedence relationships (o, o'), where task o must be performed before task o'. Each task o of product family p in generation g has a processing time pt^g_{oep} if performed by equipment e. We consider a line in which products traverse sequentially through multiple stations $\mathcal{S} = \{1 \dots S\}$, with products entering in arbitrary orders and progressing through tasks with negligible setup times. We denote the takt time as C, which is given, and identical for all stations as defined in a paced line.

Each station has a single resource (either a human worker or a robot) with several equipment pieces. Each task can be performed either by a worker or by a robot. We denote by $\mathcal{R}$ the set of resources and by $\mathcal{E}$ the set of equipment. Furthermore, the compatibility between tasks, resources, and equipment is defined by sets CR_e and CE_o that respectively contain eligible equipment for each task o and certified resources for operating such equipment e.

The primary objective is to design a reconfigurable assembly line capable of adapting to diverse product generations, efficiently. Reconfiguration involves rearranging resources and equipment, each action incurring associated costs such as purchasing, selling, installation, and uninstallation. Each of these reconfigurations is associated with a cost: α_{eg} and α'_{rg} denote the purchasing cost of equipment e and robot r (or hiring cost of worker r) in production generation g, respectively. β_{eg} and β'_{rg} denote the selling price of equipment e and robot r in production generation g, respectively. λ_{eg} and λ'_{rg} are the installation cost of equipment e and robot r in production generation g, respectively. Finally, γ_{eg} and γ'_{rg} are the un-installation cost of equipment e and robot r in production generation g, respectively. These costs vary across production gen-

erations and entail careful planning to minimize overall expenses while ensuring the assembly line's readiness for evolving product families.

A robust scenario-based mixed-integer linear programming ($MILP$) has been developed in Hashemi-Petroodi et al. (2022), named as $MILP^{Ro}$. $MILP^{Ro}$ aims to minimize the total design and future reconfiguration cost for the worst-case scenario, denoted as Y, among all sample scenarios created in a tree. An approach has been developed which describes how the scenario tree is built for the evolution of product families over several periods during the life cycle of the line. Hashemi-Petroodi et al. (2022) denoted the set of scenarios by $SC = \{1 \dots N\}$, where each scenario $n \in SC$ is a succession of pairs (p, p') of product families such that production moves from family $p \in PS_{g-1}$ to family $p' \in PS_g$ in the next generation. In $MILP^{Ro}$, decisions on task, equipment and resources allocation to stations were made for each product family p at each generation g for all scenarios n. The objective function of $MILP^{Ro}$ is min Y, where $Y \geq Q_n + Q'_n + Z_n + Z'_n \;\; n \in SC$. Q_n is the purchase/selling cost of the equipment, Q'_n is the purchase/selling cost of the robots, and the cost of hiring/firing the workers, Z_n is the installing/uninstalling cost of the equipment, and Z'_n shows is the installing/uninstalling cost of the robots and workers. For more details of the $MILP^{Ro}$, we refer to Hashemi-Petroodi et al. (2022).

However, in this study, we aim to develop an adversarial approach (AA) that accelerates the discovery of the worst-case scenario. We do not consider the scenario tree. Precisely, the evolved product families must be created and they are not given as in the $MILP^{Ro}$. Therefore, we modify the decision variables considered in $MILP^{Ro}$, accordingly. The details of the AA is given below.

4. ADVERSARIAL APPROACH (AA)

This section introduces a robust optimization approach, distinct from the developed $MILP^{Ro}$, which computes the worst-case scenario based on theoretical analysis rather than sampling scenarios in Hashemi-Petroodi et al. (2022). Employing an adversarial approach (AA) to accelerate the discovery of the worst-case scenario, the robust solution outperforms the $MILP^{Ro}$ in terms of speed and efficiency, despite yielding a higher cost. The adversarial technique, known for its effectiveness in solving robust optimization problems, operates through a two-step process comprising the master problem (MP) and the sub-problem (SP). Initially, the MP is solved considering a finite and small set of scenarios to optimize decisions accordingly (Browne et al., 1988). Subsequently, the SP identifies a scenario rendering the MP's solution infeasible, thereby enhancing solution robustness, iteratively. Note that each SP targets one scenario per generation, determining task processing times and precedence relationships to maximize product variant complexity, thereby enhancing cost-effectiveness. By iteratively refining the solution through the master and sub-problems, the approach aims to strike a balance between minimizing total costs and optimizing task, resource, and equipment assignments/reconfigurations for each generation.

Master problem (MP): First, we create a small scenario tree (named as SC') and solve the proposed $MILP^{Ro}$. After getting the solution of the MP, the design and reconfiguration of the lie for all generations are given. Then, we keep only the initial design of the line including the resource and equipment assignment to stations at $g = 0$. Let's consider binary variables w^g_{sr} is equal to 1 if resource r is assigned to station s in generation g, and 0 otherwise; x^g_{so} is equal to 1 if task o is performed at station s during production generation g, and 0 otherwise; b^g_{se} is equal to 1 if equipment e is installed at station s in generation g, and 0 otherwise; and b^g_{soe} is equal to 1 if equipment e is installed at station s to perform task o in generation g, and 0 otherwise. Then, after solving the MP, we fix the decision variables w^0_{srp}, b^0_{sep} and b^0_{soep} and rename them as w^{*0}_{sr}, b^{*0}_{se} and b^{*0}_{soe}, respectively. Then, we optimize such decisions for next generations via solving the SP, separately for each generation.

Sub-problem (SP): For every generation $g \in G - g = 0$, we formulate an SP that determines the worst product model adding to the line (the worst scenario), only optimizes one generation forward, and solves $MILP_{sub}$ (given below), which is adapted from the $MILP^{Ro}$. The initial design of the line is fixed. The SP ($MILP_{sub}$) is solved in two rounds for the first and second generations. Given the intiial design for $g = 0$ from the MP, for every new generation ($|G| = 1$), each SP looks for exactly one situation $n = 1$ as the worst product family $|PS_g| = 1$. Indeed, the design of the line from the MP is given by w^{*0}_{sr}, b^{*0}_{se} and b^{*0}_{soe}, and the SP optimizes the reconfiguration of the first generation $g = 1$ and fix them as w^{*1}_{sr}, b^{*1}_{se} and b^{*1}_{soe}, then SP optimizes the reconfiguration of the second generation $g = 2$, and it continues for next generations. We add this scenario to set SC' after producing the entire scenario (for all generations), and the new scenario tree solves the MP. As a result, the SP's variables specify the precedence graph, task processing times, and the total number of tasks needed for the new product family.

The $MILP_{sub}$ is crated. The worst-case scenario for the number of tasks needed is when the new product family needs the greatest number of tasks that can be completed, which is the upper bound of the suggested interval given by the planner/expert. The present product family at $g = 0$ requires a set of tasks is denoted by O', and after evolution of the family, the new tasks added to the joint precedence graph in the new generation which is denoted by the set O''. As a result, we define the following decision variables to ascertain the processing time and task precedence relationships for $g = 1$ and $g = 2$:

- $M_{oo'}$ is equal to 1 if task $o \in O' \cup O''$ preceeds task $o' \in O''$, and 0 otherwise.
- $pt'_{oe}{}^{g} \geq 0$ is the processing time of task $o \in O' \cup O''$ in the single scenario (product family) generated at the generation g.

These variables are defined in the $MILP_{sub}$ because the SP looks for a family of products that produces the worst-case scenario. Typically, in such scenarios, most tasks have a long processing time, and the precedence structure has a large degree. Once the MP has been solved, the SP seeks to determine, for each generation, the worst-case of the product family that maximizes the product variant complexity,

which is equivalent to maximizing the cost. The model search for the product process time ${pt'_{oe}}^{g}$ and precedence relationships $M_{oo'}$ that maximizes the costs. Conversely, the MP seeks to minimize costs and balance the line design for every generation. The $MILP_{sub}$'s objective function is (1), where $F(x, w, b, pt', M)$ is a function of the variables that are already present in the model.

$$\max_{pt', M} \Big(\min \; F(x, w, b, pt', M) \Big) \tag{1}$$

It is challenging to include this function in the SP. For instance, a local search algorithm could be developed for the Max-part of the model. The inner Min-part of the model might then be optimized using $MILP_{sub}$ to evaluate the scenarios. However, in the AA, addressing the SP using a combined strategy of mathematical programming and local search is inefficient. Therefore, rather than using the Max-Min function, we simplify it to a Max-Max one. The primary goal of this conversion is to effectively provide an approximation of the solution. In fact, the process designers re-design the line after watching the new product family, and the Max-Min model finds a robust solution for the real worst-case scenario. Though it tries to balance costs, the Max-Max function only yields a solution that corresponds to the worst-case scenario of the product variants. Because the function only affects the SP and is only utilized to provide the worst scenario for the collection of existing scenarios in the tree, this approximation has little effect on our results. The MP then attempts to maximize the overall expense in the worst-case scenario. Therefore, the function (1) is transformed to (2).

$$\max_{pt', M} \Big(\max \; F(x, w, b, pt', M) \Big) \tag{2}$$

The mathematical model $MILP_{sub}$ is finally given as (4) - (25). Note that, $MILP_{sub}$ is for a single generation with a given design for the previous generation. The objective function (4) is adapted based on (2) and aims to find the most costly product family. Since we do not consider the scenario tree, decision variables Q_n, Q'_n, Z_n and Z'_n are transformed to Q, Q', Z and Z', respectively. The value of continuous variables Q, Q', Z and Z' is calculated by equations (5), (6), (7), and (8), respectively, where Q is the purchase/selling cost of the equipment, Q' the purchase/selling cost of the robots and hiring/firing the workers, Z is the (un-)installing cost of the equipment, and Z' is the (un-)installing cost of the robots and workers.

Constraints (9) to (16) determine the re-arrangements of equipment and resources in the line when the line switches from one generation to the next. Precisely, we define some binary variables related to the equipment reconfiguration: b_e^{+g} is 1 if new equipment e is needed to be added to the line in generation g, otherwise 0; b_e^{-g} is 1 if equipment e is needed to be removed from the line in generation g, otherwise 0; $b_{se}^{\prime+g}$ is 1 if equipment e is needed to be moved to station s in generation g, otherwise 0; $b_{se}^{\prime-g}$ is 1 if equipment e is needed to be removed from station s in generation g, otherwise 0. Similarly, some binary variables for resource reconfiguration are defined as w_r^{+g}, w_r^{-g}, $w_{sr}^{\prime+g}$, $w_{sr}^{\prime-g}$. Constraints (17) ensure that each task o is performed in only one station s at generation g. Constraints (18) and (19) ensure that equipment e/ resource r is assigned at only one station s in generation g. Constraints (20) assign only one worker/robot at each station s. Constraints (21) ensure equipment is assigned to the station when performing a task there. Constraints (22) identify a compatible resource (worker/robot) at a station when the task is performed with the required equipment at that station. Constraints (23) determine the value of b_{se}^{g} according to the value of b_{seo}^{g}. Note that the model does not forbid keeping the possible stations free in the line to enhance the line's flexibility as needed.

Constraint (24) is quadratic which needs to be linearized. So for that, we give a finite upper bound M for ${pt'_{oe}}^{g}$, $o \in \mathcal{O}' \cup O'', e \in CE, g \in \mathcal{G}, |G| = 1$. Then this constraint is linearized by using the so-called big M method. We introduce a new variable $\pi_{soe}^{g} = {pt'_{oe}}^{g} b_{soe}^{g}$ with $s \in \mathcal{S}, o \in \mathcal{O}' \cup O'', e \in CE, g \in \mathcal{G}, |G| = 1$. Note that the product that we model by π_{soe}^{g} equals zero if $b_{soe}^{g} = 0$ but π_{soe}^{g} can take any value in range of 0 and M if $b_{soe}^{g} = 1$. This can be modeled using $\pi_{soe}^{g} \leq b_{soe}^{g} M$. Next, the product is always positive and smaller than ${pt'_{oe}}^{g}$, thus $\pi_{soe}^{g} \geq 0$ and $\pi_{soe}^{g} \leq {pt'_{oe}}^{g}$ with $s \in \mathcal{S}, o \in \mathcal{O}' \cup O'', e \in CE, g \in \mathcal{G}, |G| = 1$. It is left to force π_{soe}^{g} to equal ${pt'_{oe}}^{g}$ in case $b_{soe}^{g} = 1$ which we obtain with constraint (3). Therefore, Constraints (24) are linearized as explained above and also equation (3).

$$\begin{aligned} &\pi_{soe}^{g} \geq {pt'_{oe}}^{g} - (1 - b_{soe}^{g}) M \\ &s \in \mathcal{S}, o \in \mathcal{O}', e \in CE, g \in \mathcal{G}, |G| = 1 \end{aligned} \tag{3}$$

Constraints (25) are activated when a precedence relationship is considered between tasks o and o', otherwise are deactivated. The variables domain constraints are considered but not shown because of the space limit.

$$\max \; Q + Q' + Z + Z' \tag{4}$$
$$\text{s.t.}$$

$$Q = \sum_{e \in \mathcal{E}} \left[\alpha_{eg-1} \left[\sum_{s \in \mathcal{S}} {b_{se}^{*}}^{g-1} \right] + \alpha_{eg} \; b_e^{+g} + \beta_{eg} \; b_e^{-g} \right] \quad g \in \mathcal{G}, |G| = 1 \tag{5}$$

$$Q' = \sum_{r \in \mathcal{R}} \left[\alpha'_{rg-1} \left[\sum_{s \in \mathcal{S}} {w_{sr}^{*}}^{g-1} \right] + \alpha'_{rg} \; w_r^{+g} + \beta'_{rg} \; w_r^{-g} \right] \quad g \in \mathcal{G}, |G| = 1 \tag{6}$$

$$Z = \sum_{e \in \mathcal{E}} \lambda_{eg-1} \sum_{s \in \mathcal{S}} {b_{se}^{*}}^{g-1} + \sum_{s \in \mathcal{S}} \sum_{e \in \mathcal{E}} \Big[\lambda_{eg} \; b_{se}^{\prime+g} - \gamma_{eg} \; b_{se}^{\prime-g} \Big] \quad g \in \mathcal{G}, |G| = 1 \tag{7}$$

$$Z' = \sum_{r\in\mathcal{R}} \lambda'_{rg-1} \sum_{s\in\mathcal{S}} {w^*_{sr}}^{g-1} + \sum_{s\in\mathcal{S}} \sum_{r\in\mathcal{R}} \Big[\lambda'_{rg}\, w'^{+g}_{sr} - \gamma'_{rg}\, w'^{-g}_{sr} \Big] \quad g\in\mathcal{G}, |G|=1 \tag{8}$$

$$\sum_{s\in\mathcal{S}} b^g_{se} - \sum_{s\in\mathcal{S}} {b^*_{se}}^{g-1} \le b^{+g}_e \quad e\in\mathcal{E}, g\in\mathcal{G}, |G|=1 \tag{9}$$

$$\sum_{s\in\mathcal{S}} {b^*_{se}}^{g-1} - \sum_{s\in\mathcal{S}} b^g_{se} \le b^{-g}_e \quad e\in\mathcal{E}, g\in\mathcal{G}, |G|=1 \tag{10}$$

$$b^g_{se} - {b^*_{se}}^{g-1} \le b'^{+g}_{se} \quad s\in\mathcal{S},\ e\in\mathcal{E}, g\in\mathcal{G}, |G|=1 \tag{11}$$

$${b^*_{se}}^{g-1} - b^g_{se} \le b'^{-g}_{se} \quad s\in\mathcal{S},\ e\in\mathcal{E}, g\in\mathcal{G}, |G|=1 \tag{12}$$

$$\sum_{s\in\mathcal{S}} w^g_{sr} - \sum_{s\in\mathcal{S}} {w^*_{sr}}^{g-1} \le w^{+g}_r \quad r\in\mathcal{R}, g\in\mathcal{G}, |G|=1 \tag{13}$$

$$\sum_{s\in\mathcal{S}} {w^*_{sr}}^{g-1} - \sum_{s\in\mathcal{S}} w^g_{sr} \le w^{-g}_r \quad r\in\mathcal{R}, g\in\mathcal{G}, |G|=1 \tag{14}$$

$$w^g_{sr} - {w^*_{sr}}^{g-1} \le w'^{+g}_{sr} \quad s\in\mathcal{S}, r\in\mathcal{R}, g\in\mathcal{G}, |G|=1 \tag{15}$$

$${w^*_{sr}}^{g-1} - w^g_{sr} \le w'^{-g}_{sr} \quad s\in\mathcal{S}, r\in\mathcal{R}, g\in\mathcal{G}, |G|=1 \tag{16}$$

$$\sum_{s\in\mathcal{S}} x^g_{so} = 1 \quad o\in\mathcal{O}'\cup O'', g\in\mathcal{G}, |G|=1 \tag{17}$$

$$\sum_{s\in\mathcal{S}} b^g_{se} \le 1 \quad e\in\mathcal{E}, g\in\mathcal{G}, |G|=1 \tag{18}$$

$$\sum_{s\in\mathcal{S}} w^g_{sr} \le 1 \quad r\in\mathcal{R}, g\in\mathcal{G}, |G|=1 \tag{19}$$

$$\sum_{r\in\mathcal{R}} w^g_{sr} \le 1 \quad s\in\mathcal{S}, g\in\mathcal{G}, |G|=1 \tag{20}$$

$$x^g_{so} \le \sum_{e\in CE_o} b^g_{soe} \quad s\in\mathcal{S}, o\in\mathcal{O}'\cup O'' g\in\mathcal{G}, |G|=1 \tag{21}$$

$$b^g_{soe} \le \sum_{r\in CR_e} w^g_{sr} \quad s\in\mathcal{S}, o\in\mathcal{O}'\cup O'', e\in CE_o, g\in\mathcal{G}, |G|=1 \tag{22}$$

$$b^g_{soe} \le b^g_{se} \quad s\in\mathcal{S}, o\in\mathcal{O}'\cup O'', e\in CE_o, g\in\mathcal{G}, |G|=1 \tag{23}$$

$$\sum_{o\in\mathcal{O}'\cup O''} \sum_{e\in CE_o} {pt'_{oe}}^g b^g_{soe} \le C \quad s\in\mathcal{S}, g\in\mathcal{G}, |G|=1 \tag{24}$$

$$\sum_{s\in\mathcal{S}} s\, x^g_{so} - \sum_{s'\in\mathcal{S}} s'\, x^g_{s'o'} \le M(1-M_{oo'}) \quad o\in\mathcal{O}'\cup O'', o'\in O'', g\in\mathcal{G}, |G|=1 \tag{25}$$

5. COMPUTATIONAL RESULTS

MILPs are solved using IBM ILOG CPLEX Optimization Studio V12.10 on a system running MS Windows 10 Pro (64 bit) with an Intel(R) Core(TM) i7-8650U CPU @ 1.90GHz 2.11 GHz processor and 32 GB of RAM.

A data generator is adapted to suit the specific requirements of the current problem as described in details in (Mezghani et al., 4957–4979). The experiments consider different numbers of stations ($S = 4, 7$), two product models ($I = 2$) with 20 and 50 tasks (that may increase in future generations), which represent the number of tasks for the current product family. Instances are determined by the 3-tuple (I, S, O), where I, S, and O denote the number of product models, stations, and tasks, respectively. Sensitivity analysis and managerial insights are also performed by altering certain model parameters. Compatibility matrices CE_o and CR_e are randomly generated, with more capable equipment being costlier than those performing fewer tasks, and automated equipment being pricier than manual ones. Cost values are uniformly distributed, considering factors like passage of time and amortization, with approval from industrial partners.

We assess the performance of the proposed AA in terms of solution quality and execution time, compared to the proposed $MILP^{Ro}$ in Hashemi-Petroodi et al. (2022). Table 1 presents two sections for varying instance sizes, indicating the number of instances solved within the time limit, the average integrality gap (in %) for unsolved instances, and the average CPU time (in s) for the MILP. Additionally, detailed average CPU times for master and sub-problems are provided, as well as the average number of iterations that the MP is resolved with an updated set of scenarios (N_{master}) for the AA. The last column indicates that fewer worse scenarios were found in the AA's SP compared to small-size cases, owing to the limit on the number of iterations with no solution improvement. Despite this, the AA achieves quicker and better solutions compared to the $MILP^{Ro}$, as reflected in the solution quality discussed in Table 2. Moreover, Table 1 highlights that the developed AA significantly outperforms the $MILP^{Ro}$ in terms of computation speed. Notably, the CPU time for the SP represents the cumulative duration of all SP iterations. Both the AA and $MILP^{Ro}$ offer approximate solutions due to the vast number of random scenarios generated in the $MILP^{Ro}$, with a subset initiating the AA, which then proceeds with additional scenarios generated under similar conditions as the $MILP^{Ro}$.

Table 1. Average computational time (s).

Size (P, S, O)	$MILP^{Ro}$		AA		
	N° solved instances	CPU time (s)	CPU time (s) MP	CPU time (s) SP	N_{master}
(2, 4, 20)	10/10	747.1	136.4	0.2	7.7
(2, 7, 50)	10/10	8356.3	1054.8	2.8	8.6

Table 2 reports the final objective function value (OFV) resulting from $MILP^{Ro}$ and AA. In the case of AA, two

distinct columns (*OFV (MP)* and *OFV (SP)*) display the final solution of the MP and the solution of the last SP solved, respectively. Due to the heuristic nature of AA, it may not yield identical values to $MILP^{Ro}$ but generally converges closely. The first Gap (%) column quantifies the average disparity between the final solution of AA (from the MP) and the optimal solution of $MILP^{Ro}$. The last Gap (%) column shows the average difference between the final solutions of the MP and last round of solving the SP.

Table 2. Solution quality of the approaches.

Size	$MILP^{Ro}$	AA		Gap (%)	Gap (%)
(P, S, O)	OFV	OFV (MP)	OFV (SP)	$MILP^{Ro}/AA$	AA (MP/SP)
(2, 4, 20)	80455.2	80285.0	79895.1	0.2	0.5
(2, 7, 50)	84323.5	81657.2	80892.0	2.9	1.0

Only for small size instances ($(2,4,20)-size$), we compare our robust model, $MALRP$, and a classical model termed $MALRP_{Classic}$. In $MALRP_{Classic}$, decisions regarding line design and reconfiguration are made independently for each production generation without considering future scenarios. The same scenario tree is considered as the one for $MALRP$. Equipment and resource design are fixed for each period, followed by optimization of reconfiguration planning for subsequent periods. The total assembly costs of the worst scenario for each generation throughout the line's life cycle are summed up.

In Table 3, our robust model $MALRP$ consistently results in notably lower costs than $MALRP_{Classic}$, while consuming more computational time. However, the proposed AA substantially reduces the CPU time required. Table 3 shows that approximately 18% of these cost savings are attributed to purchasing/selling costs of robots and workers, and installation/un-installation costs of equipment and resources. Conversely, $MALRP_{Classic}$ saves about 4.8% on the purchasing/selling costs of equipment (Q) across all instances, although the total average of such costs in $MALRP_{Classic}$ remains slightly higher than $MALRP$. This divergence occurs because the robust model invests more, approximately 20% more than $MALRP$, in the design cost of the line by purchasing equipment and resources ($Q+Q'$), and subsequently, in the reconfiguration of the line through (un-)installation costs of equipment and resources ($Z+Z'$), particularly for equipment.

Table 3. $MALRP$ Vs. $MALRP_{Classic}$.

Problem	Q	Q'	Z	Z'	CPU time (s)
$MALRP$	9816.0	61022.6	1866.0	8377.2	747.1
$MALRP_{Classic}$	9972.2	78803.6	2730.6	12237.6	180.9

6. CONCLUSION

This study addresses the Mixed-Model Assembly Line Reconfiguration Planning Problem (MALRP) commonly encountered in the automotive industry. Given the costly nature of line design and reconfiguration, this research delves into robust optimization under uncertain future product family evolution, a novel aspect in assembly line design literature. An Adversarial Approach (AA) is developed to solve large instances in a reasonable amount of time. We compare this approach with the Mixed-Integer Linear Programming (MILP) model from Hashemi-Petroodi et al. (2022). This MILP model minimizes total costs associated with initial design and future reconfigurations, utilizing a scenario tree for future product family realizations, generated via random scenario sampling. Computational experiments conducted on benchmark instances demonstrate the superiority of the AA in terms of speed and efficiency over the MILP, especially for smaller instances where it provides solutions with lower cost values. The resulting robust model significantly reduces design and reconfiguration costs compared to the classical model. In the classical model, the line's design and reconfiguration are optimized periodically without considering future generations, leading to higher costs in addressing worst-case scenarios. Some future research can be done on the development of efficient algorithms for solving larger instances optimally and enhancing the generic approach for generating scenario trees by characterizing product families. Moreover, more cases and instances (larger size) can be tested in future to provide stronger and more generalized insights.

REFERENCES

Biswas, S., Chakrabortty, R.K., Turan, H.H., and Elsawah, S. (2023). Consideration of uncertainties in a dynamic modeling system integrated with a deep learning based forecasting approach. *CIRP Journal of Manufacturing Science and Technology*, 44, 27–44.

Browne, J., Harhen, J., and Shivnan, J. (1988). *Production management systems: a CIM perspective*. Addison Wesley Publishing Company.

Bryan, A., Hu, S.J., and Koren, Y. (2013). Assembly system reconfiguration planning. *Journal of Manufacturing Science and Engineering*, 135(4).

Hashemi-Petroodi, S.E., Thevenin, S., and Dolgui, A. (2022). Mixed-model assembly line design with new product variants in production generations. *IFAC-PapersOnLine*, 55(10), 25–30.

Koren, Y., Heisel, U., Jovane, F., Moriwaki, T., Pritschow, G., Ulsoy, G., and Van Brussel, H. (1999). Reconfigurable manufacturing systems. *CIRP Annals*, 48, 2.

Liu, M., Liu, X., Chu, F., Zheng, F., and Chu, C. (2020). Robust disassembly line balancing with ambiguous task processing times. *International Journal of Production Research*, 58(19), 5806–5835.

Manzini, M., Unglert, J., Gyulai, D., Colledani, M., Jauregui-Becker, J.M., Monostori, L., and Urgo, M. (2018). An integrated framework for design, management and operation of reconfigurable assembly systems. *Omega*, 78, 69–84.

Mezghani, Y., Hashemi-Petroodi, S.E., Thevenin, S., and Dolgui, A. (4957–4979). Robust optimization, mixed model assembly line, reconfigurability, product family evolutions, adversarial approach. *International Journal of Production Research*, 62(13), 1–23.

Sivasankaran, P. and Shahabudeen, P. (2017). Comparison of single model and multi-model assembly line balancing solutions. *International Journal of Computational Intelligence Research*, 13(8), 1829–1850.

Wei, W., Ji, J., Wuest, T., and Tao, F. (2017). Product family flexible design method based on dynamic requirements uncertainty analysis. *Procedia CIRP*, 60, 332–337.

Zhang, Z., Tang, Q., Chica, M., and Li, Z. (2023). Reinforcement learning-based multiobjective evolutionary algorithm for mixed-model multimanned assembly line balancing under uncertain demand. *IEEE Transactions on Cybernetics*.