UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E INFORMÁTICA INDUSTRIAL

ADALBERTO SATO MICHELS

INDUSTRIAL ASSEMBLY LINES WITH MULTI-OPERATED WORKSTATIONS: APPLICATIONS AND METHODS

TESE

CURITIBA
2020

ADALBERTO SATO MICHELS

INDUSTRIAL ASSEMBLY LINES WITH MULTI-OPERATED WORKSTATIONS: APPLICATIONS AND METHODS

Linhas De Montagem Industriais Com Estações De Trabalho Multi-Operadas: Aplicações E Métodos

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial, da Universidade Tecnológica Federal do Paraná (UTFPR), como requisito parcial para obtenção do título de "Doutor em Ciências" - Área de Concentração: Engenharia de Automação e Sistemas.

Orientador: Prof. Dr. Leandro Magatão

CURITIBA

2020

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Curitiba

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ADALBERTO SATO MICHELS

INDUSTRIAL ASSEMBLY LINES WITH MULTI-OPERATED WORKSTATIONS: APPLICATIONS AND METHODS

Trabalho de pesquisa de doutorado apresentado como requisito para obtenção do título de Doutor Em Ciências da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Engenharia De Automação E Sistemas .

Data de aprovação: 17 de Dezembro de 2020

Prof Leandro Magatao, Doutorado - Universidade Tecnológica Federal do Paraná

Prof Arinei Carlos Lindbeck Da Silva, Doutorado - Universidade Federal do Paraná (Ufpr)

Prof Leandro Callegari Coelho, Doutorado - Laval University

Prof.a Maristela Oliveira Dos Santos, Doutorado - Universidade de São Paulo (Usp)

Prof Ricardo Luders, Doutorado - Universidade Tecnológica Federal do Paraná

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my parents: Solange and Nelson. To my mother, for teaching me discipline and for the constant encouragement to always keep learning. To my father, for demonstrating the importance of hard work and for always being proud of me and each of my achievements.

A special thanks to my advisor, Professor Leandro Magatão, for the hours of dedication in meetings (at rather inconvenient times), work reviews, and guidance throughout the PhD degree. He certainly gave me all the support I needed for my quite ambitious plans and I can only be thankful for that.

My sincere thanks to my friends, who make everything worthwhile at all times. Eduardo and Samir, for these 15 years of friendship filled with countless bad taste jokes. Each of you is a brother from a different mother. Juliano and Tales, for the companionship in all good and bad moments of university (and now adult) life. You make every stage of my existence more enjoyable. Adriane, Alysson, Daniel, Celso, Cesar, Eduardo, Fabiane, Jan, Jéssica, Juliano, Julio, Luiza, Nathália, Pınar, Renato, Samir, Sophie, Tales, Thiago, and Vivek, for life or work related advice, research collaborations, daily love, emotional support, and for always bringing an optimistic point of view into every situation. Your company is truly precious in trips, festivals, bike rides, night-outs at weekends, virtual meetings, or simply coffee breaks. I own you all my most memorable moments, sanity, and valuable (or meaningless) discussions. I am extremely happy and privileged to have all of you in my life.

Lastly, I am thoroughly grateful for the opportunity to study at UTFPR/CPGEI and for the 2-year financial support under the Agreement 041/2017 (FA–UTFPR–RENAULT) from Fundação Araucária and Renault do Brasil during my doctorate studies.

“First say to yourself what you would be; and
then do what you have to do.”
—Epictetus, *Discourses: Book III*, c. 108.

RESUMO

MICHELS, Adalberto Sato. **Industrial Assembly Lines with Multi-operated Workstations: Applications and Methods**. 2020. 151 f. Tese (Doutorado em Engenharia Elétrica e Informática Industrial) – Universidade Tecnológica Federal do Paraná. Curitiba, 2020.

As linhas de montagem estão amplamente presentes na indústria de fabricação automotiva. O procedimento de construção de veículos emprega vários trabalhadores ou robôs equipados com um conjunto diversificado de ferramentas. Instalações, salários, robôs e ferramentas possuem altos custos, fazendo surgir a necessidade de se projetar uma linha eficiente cuidadosamente. É crucial que a demanda do produto seja atendida e que as despesas sejam reduzidas ao mesmo tempo. Devido aos produtos encontrados nas indústrias automotivas serem de grande porte, vários funcionários podem ser designados à mesma estação de trabalho para executar diferentes operações simultaneamente no mesmo produto. Nesta tese de doutorado, três estudos propostos são apresentados e discutidos: o problema do Projeto de Linha de Montagem Robótica (PLMR) e duas variantes do Problema de Balanceamento de Linha de Montagem Multi-operada (PBLMM). Formulações de Programação Linear Inteira Mista (PLIM) são desenvolvidas para todos os problemas, visando a minimização dos custos totais na taxa de produção desejada ou do tempo de ciclo dado recursos limitados. Para o problema de PLMR, várias características práticas são levadas em consideração, testes computacionais são conduzidos, e um estudo de caso prático é resolvido com parâmetros reais coletados de uma linha de montagem robotizada para soldagem, chegando à otimalidade. Em segundo lugar, ambos os modelos das variantes do PBLMM incorporam restrições fortes de quebra de simetria (inequações válidas) e decompõem o problema original em novos Algoritmos de Decomposição de Benders (ADB). Estes algoritmos são capazes de resolver instâncias grandes de forma otimizada e superar os métodos anteriores em termos de qualidade da solução. Finalmente, as contribuições destes trabalhos desenvolvidos são resumidas e direções de novas frentes de pesquisa são sugeridas para todos os problemas.

Palavras-chave: Pesquisa operacional. Otimização. Balanceamento de linhas de montagem. Programação linear inteira mista. Métodos de decomposição.

ABSTRACT

MICHELS, Adalberto Sato. **Industrial Assembly Lines with Multi-operated Workstations: Applications and Methods**. 2020. 151 p. Thesis (PhD in Electrical and Computer Engineering) – Universidade Tecnológica Federal do Paraná. Curitiba, 2020.

Assembly lines are widely present in the automotive manufacturing industry. The procedure of building vehicles employs several workers or robots equipped with a diverse pool of tools. Facilities, wages, robots, and tools are quite costly, giving rise to the necessity of consciously designing an efficient line. It is crucial to meet product demand and reduce expenses at the same time. Due to large-size products found in automotive industries, multiple workers can be assigned to the same workstation in order to simultaneously perform different operations on the same product. In this PhD thesis, three proposed studies are presented and discussed: the Robotic Assembly Line Design (RALD) problem and two variants of the Multi-manned Assembly Line Balancing Problem (MALBP). Mixed-Integer Linear Programming (MILP) formulations are developed for all three problems. They either aim at minimising total costs at the desired production rate or the cycle time given limited resources. For the RALD problem, several practical characteristics are taken into consideration, computational tests are conducted, and a practical case study is solved with parameters collected from a real-world robotic welding assembly line, reaching optimality. Secondly, both variants of MALBP models incorporate strong symmetry break constraints (valid inequalities) and decompose the original problem into innovative Benders' Decomposition Algorithms (BDA). These algorithms are able to optimally solve large-size instances and outperform previous methods in terms of solution quality. Finally, contributions of developed works are summarised and further research directions are suggested for all problems.

Keywords: Operational research. Optimisation. Assembly line balancing. Mixed integer linear programming. Decomposition methods.

LIST OF ALGORITHMS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ACO	Ant Colony Optimisation
ALBP	Assembly Line Balancing Problem
ALDP	Assembly Line Design Problem
ALWABP	Assembly Line Worker Assignment and Balancing Problem
B&B	Branch-and-Bound
BDA	Benders' decomposition algorithm
CBC	Combinatorial Benders' Cut
CT	Cycle Time
DSA	Direct Simulated Annealing
DT	Dead Time
EFO	Electromagnetic Field Optimisation
GA	Genetic Algorithm
GALBP	General Assembly Line Balancing Problem
GSA	Gantt Simulated Annealing
HT	Hash Table
ISA	Indirect Simulated Annealing
LB	Lower Bound
MALBP	Multi-manned Assembly Line Balancing Problem
MILP	Mixed-Integer Linear Programming
MP	Master Problem
OR	Operational Research
PALBP	Parallel Assembly Line Balancing Problem
PAM	Parallel Assignment Method
PSO	Particle Swarm Optimisation
RALB	Robotic Assembly Line Balancing
RALD	Robotic Assembly Line Design
RSW	Resistance Spot Welding
SA	Simulated Annealing
SALBP	Simple Assembly Line Balancing Problem
SH	Simplification Hypotheses
SP	Slave Problem
TALBP	Two-sided Assembly Line Balancing Problem
UB	Upper Bound
VWALBP	Assembly Line Balancing Problem with Variable Workplaces

CONTENTS

1 INTRODUCTION

The automotive industry is a segment in which both manual and robotic assembly lines are widely applied. Such lines are designed and implemented with months of planning in advance; however, the line's configuration is rarely guided by modelling and solution approaches provided by the Operational Research (OR) techniques. The optimisation conducted by OR seeks, for instance, to abstract as faithfully as possible real-world problems using mathematical models. Then, in order to obtain the best answer for several configurations of systems or operations, OR methods attempt to solve these problems by employing computational approaches.

For "best answer", one can take into account several factors: the greatest efficiency or productivity of a system or operation, the lowest cost or time to carry out a project, among others. All of that depends on what criteria are being considered at the time the problem will be solved. Currently, the vast majority of industrial systems operate under sub-optimal conditions. For a long-term application proposal, therefore, one should formally represent the available resources a company has to reach a given goal in a mathematical model. Once this model is solved, its solution has to be interpreted, resulting in insights that make one capable of responding which is the best system design to be configured in order to meet the desired requirements. The optimised design is a fundamental issue to allow an industrial system to operate in an optimal condition, according to the adopted criterion.

The process of designing and balancing production lines is present at a strategic level of global decisions in industries. Before the implementation of a new line, several studies are carried out and a large amount of data is collected, including the probable installation costs by consulting suppliers' prices, the available physical space to build this new line, market demand surveys are conducted for the new product, among other factors. After this step, the design of the line actually begins, which could be guided by an optimisation process that uses the previously collected data. This optimisation process consists in modelling the industrial characteristics and solving the model by computational methods. The obtained solution finds the best manner to allocate resources: capital, time, etc. Thus, the optimal answer in this scenario would be the one that allows the minimisation of costs for the line design, while providing the aimed productivity. Alternatively, it is also possible to pursue productivity maximisation for a given limited budget or finite resources. Furthermore, the size of products to be assembled can be crucial to decide if multiple workers or robots are able to simultaneously perform the necessary activities on the

same product, whilst sharing a predetermined area.

This PhD thesis focuses on presenting developed research works applied to the automotive industry, where there is already a research effort for possible applications of optimisation methods for the production line reconfiguration (Chapter 2). Nonetheless, the studies carried out herein apply more specifically to the design of robotic welding assembly lines and balancing multi-manned assembly lines, due to the fact that these are fields that still lacked studies, which leads to a gap between literature and practical problems, as detailed in Chapter 2. These features might be combined and are commonly found together in real-world applications, as illustrated in Figure 1, in which multiple welding robots perform tasks simultaneously on the same work-piece.

Figure 1 – Example of a multi-operated robotic welding assembly line: robots holding spot welding tools (1) and a vehicle's body (2) are illustrated.

Source: Michels (2017).

Some factors contribute to this gap. The several welding tasks is one them, which imply both in the possibility of leaner arrangements in the line and physical limitations in the assembly order. The use of different robots and tools for each assembly stage, the movement of the work-pieces to be produced, and the difficulties in measuring the trade-off between costs and productive efficiency in the robotic assembly line are also practical characteristics that make the problem hard to be modelled. These extensions are further discussed in Chapter 3.

Conversely, the use of multi-manned assembly lines also implies in balancing difficulties: since tasks cannot be performed in an arbitrary order, one needs to cope with task scheduling problems for workers that perform activities in the same workstation. Therefore, the goal must consider the number of workers and stations used in the final configuration, as well as satisfying minimal production rate requirements and technological constraints. This additional flexibility is further presented and analysed in Chapter 4. Alternatively, striving for the maximal efficiency in the production rate can also be a goal. In that case, the resource limitation is given by the available workforce. Such alternate scenario is further explored in Chapter 5.

Thus, two classes of problems are studied throughout this PhD thesis: the Robotic Assembly Line Design (RALD) problem and the Multi-manned Assembly Line Balancing Problem (MALBP). The first problem's main idea consists in designing a fully automated assembly line at the lowest cost, taking into account cost parameters and practical features found in the automotive industry, so that the model generates answers that guide the implementation of new configurations. The second one contemplates two problem variants of assembly systems in which products are relatively large in relation to the manual or robotic workers, hence allowing multiple tasks to be concomitantly performed by different workers. For the former variant of this problem, the objective is to find the configuration that requests the least number of workers and stations, whilst respecting demand rates. The latter variant attempts to achieve maximal output rate, while considering a limited number of workers. Task scheduling must be considered in each station for both variants.

1.1 OBJECTIVES

The main objective of this PhD thesis is to propose practical extensions regarding the RALD (Robotic Assembly Line Design) problem and the MALBP (Multi-manned Assembly Line Balancing Problem) to bridge gaps found in the literature. The specific objectives are detailed as follows in Section 1.1.1.

1.1.1 Specific Objectives

- Assess the scientific and industrial value of tackling RALD and MALBP, identifying possible literature gaps;
- Present the RALD problem: propose a mathematical model with practical extensions and

solve real-world problems;

- Present the type-1 MALBP: propose a mathematical model with symmetry break constraints and a solution method based on Benders' decomposition algorithm;
- Present the type-2 MALBP: extend the mathematical model for the type-2 problem and propose an exact algorithm approach based on initial solution procedures and combinatorial Benders' cuts; and
- Identify limitations of the proposed models and solution strategies for RALD and MALBP in order to provide directions for future research.

1.2 RESEARCH JUSTIFICATION AND LIMITATIONS

Assembly lines applied to industries that build large products are quite capital-intensive (e.g. car manufacturing companies). Optimisation techniques may help alleviate the burden of some costs or get the most out of a system in operation. In that regard, this work aims to contribute with strategies to tackle problems found in automotive industries, which naturally arise from practical applications.

More specifically, the approaches herein proposed attempt to (i) minimise designing costs of robotic lines, (ii) minimise operating resources in multi-operated manual lines to meet a given demand, or (iii) maximise the productivity of multi-operated manual lines given a workforce restriction.

Since all proposed solution methods are exact algorithms, limitations are expected. Due to the combinatorial nature of the problems herein addressed, shortcomings may arise regarding the time required to obtain optimal solutions in sizable or pathological cases.

1.3 PUBLICATIONS

This PhD thesis and the following references are the results of the author's PhD studies and research. The content of journal articles from Section 1.3.1 are the basis for Chapters 3, 4, and 5 of this thesis.

1.3.1 Journal articles

- MICHELS, Adalberto Sato; LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. The Robotic Assembly Line Design (RALD) problem: Model and case studies with practical extensions. **Computers & Industrial Engineering**, v. 120, p. 320–333, 2018.

- MICHELS, Adalberto Sato; LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. A Benders' decomposition algorithm with combinatorial cuts for the multi-manned assembly line balancing problem. **European Journal of Operational Research**, v. 278, n. 3, p. 796–808, 2019.

- MICHELS, Adalberto Sato; LOPES, Thiago Cantos; MAGATÃO, Leandro. An exact method with decomposition techniques and combinatorial Benders' cuts for the type-2 multi-manned assembly line balancing problem. **Operations Research Perspectives**, v. 7, p. 100163, 2020.

1.3.2 Co-authored journal articles

- LOPES, Thiago Cantos; MICHELS, Adalberto Sato; SIKORA, Celso Gustavo Stall; MOLINA, Rafael Gobbi; MAGATÃO, Leandro. Balancing and cyclically sequencing synchronous, asynchronous, and hybrid unpaced assembly lines. **International Journal of Production Economics**, v. 203, p. 216–224, 2018.

- LOPES, Thiago Cantos; MICHELS, Adalberto Sato; MAGATÃO, Leandro. A note to: A hybrid algorithm for allocating tasks, operators, and workstations in multi-manned assembly lines. **Journal of Manufacturing Systems**, v. 52, p. 205–208, 2019.

- LOPES, Thiago Cantos; MICHELS, Adalberto Sato; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. Balancing and cyclical scheduling of asynchronous mixed-model assembly lines with parallel stations. **Journal of Manufacturing Systems**, v. 50, p. 193–200, 2019.

- LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; MAGATÃO, Leandro. Mixed-model assembly lines balancing with given buffers and

product sequence: model, formulation comparisons, and case study. **Annals of Operations Research**, v. 286, p. 475–500, 2020.

- LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; MAGATÃO, Leandro. An iterative decomposition for asynchronous mixed-model assembly lines: combining balancing, sequencing, and buffer allocation. **International Journal of Production Research**, v. 58, n. 2, p. 615–630, 2020.

- LOPES, Thiago Cantos; PASTRE, Giuliano Vidal; MICHELS, Adalberto Sato; MAGATÃO, Leandro. Flexible Multi-Manned Assembly Line Balancing Problem: Model, Heuristic Procedure, and Lower Bounds for Line Length Minimization. **Omega**, v. 95, p. 102063, 2020.

- LOPES, Thiago Cantos; MICHELS, Adalberto Sato; LÜDERS, Ricardo; MAGATÃO, Leandro. A simheuristic approach for throughput maximization of asynchronous buffered stochastic mixed-model assembly lines. **Computers and Operations Research**, v. 115, p. 104863, 2020.

1.3.3 Conference proceedings

- LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; MAGATÃO, Leandro. A New Model for Simultaneous Balancing and Cyclical Sequencing of Asynchronous Mixed-Model Assembly Lines with Parallel Stations. *In*: **XLIX Simpósio Brasileiro de Pesquisa Operacional**. Blumenau - SC: SBPO, 2017. p. 1–12.

- SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; LOPES, Thiago Cantos; MAGATÃO, Leandro. Combining k-Opt Improvement Procedure and Tabu-Search to Solve the Symmetric TSP via MILP Formulation. *In*: **XLIX Simpósio Brasileiro de Pesquisa Operacional**. Blumenau - SC: SBPO, 2017. p. 3533.

- MICHELS, Adalberto Sato; LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. A new mathematical formulation with search-space reduction techniques for the multi-manned assembly line balancing problem. *In*: **29th European Conference on Operational Research**. Valencia: EURO, 2018. p. 135.

- LOPES, Thiago Cantos; MICHELS, Adalberto Sato; SIKORA, Celso Gustavo Stall; SILVA, Arinei Carlos Lindbeck; MAGATÃO, Leandro. Modeling the Stochastic Steady-State of Mixed-Model Asynchronous Assembly Lines with Markov Chains. *In*: **XIX Latin-Iberoamerican Conference on Operations Research**. Lima - Peru: CLAIO, 2018. p. 183–190.

- LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; MAGATÃO, Leandro. A Matheuristic for Makespan Minimization of Asynchronous Stochastic Mixed-Model Assembly Lines. *In*: **L Simpósio Brasileiro de Pesquisa Operacional**. Rio deJaneiro - RJ: SBPO, 2018. p. 1–12.

- MICHELS, Adalberto Sato; SIKORA, Celso Gustavo Stall; LOPES, Thiago Cantos; MAGATÃO, Leandro. An MILP Formulation for Local Search k-Opt Improvement Procedures. *In*: **L Simpósio Brasileiro de Pesquisa Operacional**. Rio de Janeiro - RJ: SBPO, 2018. p. 1–12.

1.4 DOCUMENT OUTLINE

This chapter introduced the readers to assembly lines commonly found in automotive industries and the problems that were researched during the author's PhD studies. Chapter 2 presents the main terminology and operational aspects existing in the assembly line balancing problem, focusing on the differences found in its variants and possible practical extensions. The goal is to initially present the main singularities involved in the studied balancing problems to readers. Further operational aspects are exploited in detail throughout Chapters 3 to 5.

Chapter 3 investigates the Robotic Assembly Line Design (RALD). This problem had initially been proposed by the author in Michels (2017). Since then, this work has gone through a peer-review process, with the inclusion of several improvements and the extension of its results. The problem's definition, model, and results are herein presented. The elements considered in the problem are illustrated and explained, advantages of parallel stations are discussed and the specificity of body-in-white stage tasks are also described. These practical characteristics are further included in the modelling formulation and applied to computational and industrial case studies. This chapter's content includes in full, with slight adaptations, the published paper Michels *et al.* (2018b): *The Robotic Assembly Line Design (RALD) problem: Model and case studies with practical extensions*, which was published in the *Computers &*

Industrial Engineering.

Chapter 4 presents the type-1 Multi-manned Assembly Line Balancing Problem (MALBP-1). Definitions, monolithic model, developed Benders' decomposition algorithm (BDA), and results for both approaches are explored. These results are compared in terms of solution quality and time to the best-known mathematical model and method for the same problem. The content of this chapter derives from the published paper Michels *et al.* (2019): *A Benders' decomposition algorithm with combinatorial cuts for the multi-manned assembly line balancing problem*, which was published in the *European Journal of Operational Research*.

Chapter 5 extends the Multi-manned Assembly Line Balancing Problem to contemplate the type-2 variant (MALBP-2). Moreover, it adapts and improves the BDA developed in the previous paper. It does so by proposing an initial solution procedure and incorporating new cuts. This chapter includes the content of the paper Michels *et al.* (2020): *An exact method with decomposition techniques and combinatorial Benders' cuts for the type-2 multi-manned assembly line balancing problem*, which was published in the *Operations Research Perspectives*.

Finally, Chapter 6 presents concluding remarks. Final considerations and main contributions of this PhD thesis are discussed. Furthermore, future research directions and possibilities are suggested for readers who may be interested in continuing the developed work.

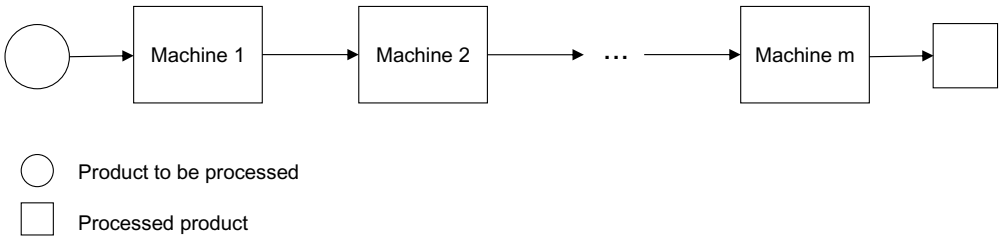
2 THE ASSEMBLY LINE BALANCING PROBLEM

This chapter introduces the main concepts associated with assembly line balancing problems, which will serve as a basis for the comprehension of the remaining chapters. It presents a bibliography review for production layouts, the (Simple and General) Assembly Line Balancing Problem, the Assembly Line Design Problem, robotic lines, and multi-operated workstations. It goes through crucial simplification hypotheses, solution methods, and problems' variations with practical extensions. Besides, stations paralleling, equipment selection, and resistance spot welding procedures are further exhibited in order to specifically define the proposed RALD problem. The feature of multiple workers sharing the same station and simultaneously performing tasks on the same product piece is also further explored to define the MALBP.

2.1 BASIC CONCEPTS FOR PRODUCTION LINES

According to Krajewski *et al.* (2013), the most usual forms of layout design used in production systems are: flow-shop, job-shop, fixed position and hybrid. Their organisations depend on the product variety, volume, and complexity, and the layout pattern is chosen for a better system efficiency. As production systems used in the automotive industry are frequently based on flow-shop layouts (PINEDO, 2016), only this design is further explored (Figure 2).

Figure 2 – Example of flow-shop.



Source: Michels (2017).

Flow-shops are considered to have a product-oriented design, i.e. the machinery and the operations are organised for the product flow (see Figure 2). This flow can either be continuous, synchronous, or asynchronous (LOPES *et al.*, 2018a; LOPES *et al.*, 2020b). Assembly lines in a flow-shop configuration are generally dedicated to produce homogeneous goods, enabling their mass production. The advantages of this production layout for regular operation are, among

others, the high system utilisation, constant and simple material flowing and handling, low setup times, and high throughput. However, unavoidable drawbacks comes along with the flow-shop configuration. Costly capital investments and line stoppages are among them. As each stage is dependent on the integral system's operation for the proper production flow (KRAJEWSKI *et al.*, 2013), the entire system has to be temporarily shut down whenever maintenance is required of a failure occurs.

Precisely because of the high costs and utilisation of the line, designing and balancing a flow-shop assembly line is a long-term decision. This fact makes it an important problem, whose solution can be aided by optimisation methods in order to achieve potential economy and better efficiency. Sequencing and scheduling problems may arise if this layout is employed to a family of similar products (LOPES *et al.*, 2019; LOPES *et al.*, 2020; LOPES *et al.*, 2020a) or if parallel work is used within workstations (MICHELS *et al.*, 2019; LOPES *et al.*, 2020; MICHELS *et al.*, 2020).

It follows that this kind of flow-shop configuration has given rise to the traditional Assembly Line Balancing Problem (ALBP). The ALBP is vastly approached in the literature, as indicated in, for instance, Scholl (1999), Scholl and Becker (2006), Becker and Scholl (2006), Boysen *et al.* (2007), and Battaïa and Dolgui (2013).

An assembly line is composed of a set of minimum rational work element (hereafter referred to as *task*) for the product assemblage. Tasks are allocated into a set of workstations (*stations*, hereafter). In order to assure the product flow from one station to another; these stations are linked together by a transport system (BAYBARS, 1986).

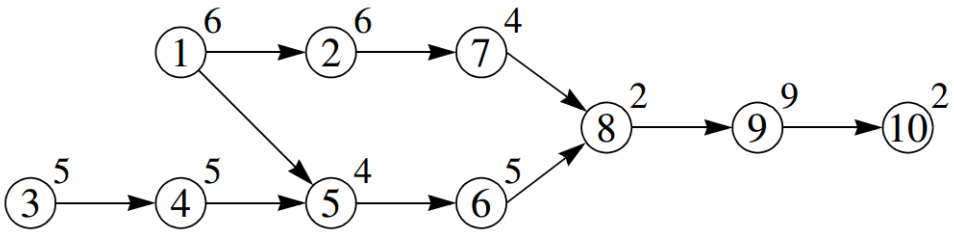
Tasks are the smallest indivisible work element. They require an amount of *processing time* to be performed, which are known as parameters. These tasks must be performed at stations by human or robot operators using specific equipment (machinery or tools).

In serial lines, the line's *cycle time* is defined by the most loaded station. The production rate is determined by calculating how much time one unit of the finished product takes to emerge from the last station along the line (SCHOLL, 1999).

Another characteristic of assembly lines is that tasks cannot be performed in an arbitrary order. They might have technological sequencing requirements, namely *precedence relations*. When all precedence relations are known, they can be schematically represented by a directed acyclic graph, namely *precedence diagram*. Figure 3 (SCHOLL; BECKER, 2006) shows a precedence diagram with 10 tasks (number inside the circle) with task processing times between

2 and 9 time units (top-right corner). Task 5 is chosen to exemplify how precedence constraints work: in order to process task 5, tasks 1, 4 (direct predecessors), and 3 (indirect predecessor) are required to be completed. On the other hand, task 5 must be finished before its direct and indirect successors (6, 8, 9, and 10) are started.

Figure 3 – Example of a precedence diagram: the number inside the circle represents the task number and at the top-right corner the task process time.



Source: Scholl and Becker (2006).

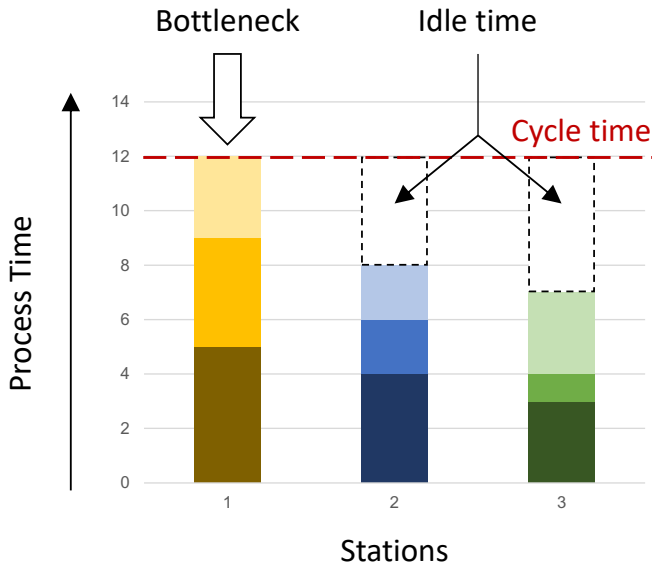
Even though the line's cycle time is defined by the most loaded station (*bottleneck*), it can be stated that each station has its own cycle time, which is defined by the sum of task processing times assigned to such station. This individual cycle time might be lower than the global cycle time. This time slack between each individual station cycle time and the bottleneck's cycle time is considered unproductive. This unproductive period is called *idle time*. Figure 4 presents three stations: the first one is the bottleneck and defines the line's cycle time (12 time units), whereas the second and third stations have idle times of 4 and 5 time units, respectively.

The line is said *balanced* if the sum of idle times of all stations along the line is as low as possible (BAYBARS, 1986). Theoretically, it is possible to achieve *perfect balancing*, which occurs when tasks can be gathered so that all station total processing times are identical. Figure 5 is an example of a perfectly balanced line of the previous sub-optimal task distribution presented in Figure 4. Nevertheless, perfect balancing is unattainable in most practical cases.

According to Baybars (1986), the assumptions that can be applied to *any* deterministic model of an Assembly Line Balancing Problem (ALBP) are stated in the following simplification hypotheses (SH):

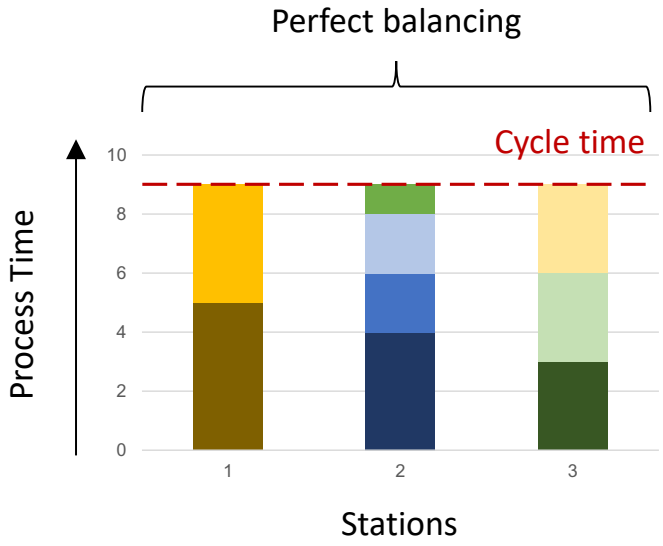
- (SH-1): all input parameters are deterministic and known;
- (SH-2): tasks are indivisible and cannot be split into two or more stations;

Figure 4 – Example of an unbalanced production system: the cycle time is defined by the bottleneck station (12 time units), whereas stations 2 and 3 have idle time. Each task is represented by a different colour.



Source: Michels (2017).

Figure 5 – Example of a perfect balancing: all three stations have the same total process time, which defines the cycle time (9 time units) and guarantees a perfectly balanced system. Each task is represented by a different colour, notice the reallocation of tasks with respect to Figure 4.



Source: Michels (2017).

- (SH-3): tasks cannot be processed arbitrary due to technological precedence restrictions; and

- (SH-4): all tasks must be processed.

A recent survey regarding ALBPs is conducted by Battaïa and Dolgui (2013).

2.2 SIMPLE ASSEMBLY LINE BALANCING PROBLEM

The ALBP has been studied by Salveson (1955) for the first time, introducing the problem. However, only after Jackson (1956) task indivisibility and precedence constraints were solved along with the problem. Bowman (1960) proposed a consistent mixed-integer linear programming (MILP) formulation, which was further enhanced by White (1961).

Along with the mathematical programming models' development, SH (5-10) were added to the previous SH (1-4) to define a new class of problem. More precisely, there is an addition of six SH in order to strictly define the Simple Assembly Line Balancing Problem (SALBP), as follow:

- (SH-5): all stations are equipped and manned to process any task;
- (SH-6): task processing times are not sequence dependent;
- (SH-7): any task can be processed at any station;
- (SH-8): the assembly line is strictly serial;
- (SH-9): the assembly line is designed for a single product; and
- (SH-10): the parameters are deterministic.

With this set of assumptions (SH-1)–(SH-10), most of the literature papers focus on achieving the best assignment of tasks among several stations arranged in a serial line, considering many of these restricting assumptions. Nonetheless, the developed research on SALBPs do not always describe and solve more realistic problems (BATTAÏA; DOLGUI, 2013).

According to Baybars (1986), these differences in describing realistic features divide the classification in two categories, making proposed problems to fall into one of the them: the Simple Assembly Line Balancing Problem (SALBP), and the General Assembly Line Balancing Problem (GALBP). The latter is further discussed in Section 2.3.

Furthermore, an extensive review on the SALBP is conducted by Scholl and Becker (2006), in which four versions of the problem are described as they arise in the literature. Table 1 divides SALBP versions according to their optimisation objective and given parameters.

Table 1 – SALBP versions based on their optimisation objective and given parameters.

	Cycle time (CT)	
	Given	Minimise
Number of stations (NS)		
Given	SALBP-F	SALBP-2
Minimise	SALBP-1	SALBP-E

Source: Scholl and Becker (2006).

The simplest problem is SALBP-F, which only establishes whether or not a feasible distribution can be achieved in a given combination of NS and CT.

In addition to (SH-1)–(SH-10), the optimisation problem in SALBP-1 receives another assumption regarding the line's CT (SH-11):

- (SH-11): the cycle time (CT) is given and fixed.

Therefore, the goal of the SALBP-1 variation is to minimise the number of stations along the line. Analogously, SALBP-2 considers (SH-1)–(SH-10) and has an additional assumption to fix NS as a parameter:

- (SH-12): the number of stations (NS) is given and fixed.

This SALBP-2 variation's goal is to minimise the cycle time or, in other words, maximise the production rate.

Lastly, SALBP-E is the most general problem version, in which the line's efficiency is maximised by simultaneously minimising CT and NS, generally considering their interrelationship in a non-linear approach ($CT \cdot NS$).

2.3 GENERAL ASSEMBLY LINE BALANCING PROBLEM

As mentioned in Section 2.2, SALBPs do not describe as many practical features as those found in the General Assembly Line Balancing Problem (GALBP). In SALBPs, the system is basically restricted by precedence relations, task distribution, cycle time and/or number of stations constraints, as indicated in (SH-1)–(SH-12). Whereas, GALBPs regard further specifications, such as task incompatibility, space constraints, parallel lines or station, multiple and capable workers, equipment selection, unproductive times, among others. These specifications imply in the relaxation of one or any combination of the previously stated simplification hypotheses. A particularly focused review on the GALBP is conducted by Becker and Scholl (2006).

This section is intended to present a concise review on some applications of GALBPs. The problem variants that are mostly related to the proposed problems in this PhD thesis are highlighted. These included features are thereby summarised in Section 2.7. Since many of these ALBP extensions are explored individually, this section also lists and discusses which SHs are being relaxed by the reviewed work. By all means, (SH-1)–(SH-4) are kept in GALBPs and so is (SH-5) for now. This last exception is later discussed in Section 2.4.

Task processing times are not sequence dependent (SH-6), i.e. they are independent of the station at which they are performed and of preceding or following tasks. The most common example of a practical extension that requires the relaxation of this hypothesis is the presence of set-up times. This can be seen as a preparation time needed between tasks (SCHOLL *et al.*, 2008; SCHOLL *et al.*, 2013), which is generally related to tool changing or task properties. Another possibility is the presence of fixed unproductive times, namely dead time. It can occur before products start processing in each station (BARD, 1989), between tasks (LOPES *et al.*, 2017), and/or after products are done processing in each station. Dead times are generally related to the robots, workers or conveyors movement time.

Any task can be processed at any station (SH-7), which means that there are no positional, layout or zoning constraints. In practical scenarios, this hardly is the case: tasks cannot always be assigned to any station (DECKRO, 1989), some set of specific tasks frequently must be allocated either at the same station (inclusion constraints), or are incompatible and must be placed at different stations (exclusion constraints). Incompatibility or zoning restriction problems might happen due to positioning or accessibility problems (ESSAFI *et al.*, 2010), fixed machinery and minimal/maximal distance between assignments (SCHOLL *et al.*, 2010), among other issues concerning task assignment restrictions (SCHOLL *et al.*, 2010; SIKORA *et al.*, 2017a).

The assembly line is strictly serial (SH-8). Therefore, processing times are additive at any station, since there are no feeder or parallel sub-assembly lines. Any possible interaction of this type is neglected. The balancing of U-shaped lines is frequently approached in the literature as an alternative to extend the SALBP. U-lines allow a different grouping of tasks in the precedence diagram, since workers may cross the line to perform tasks at the beginning and at the end of the line, providing better, or at least equal, results when compared to serial lines (MILTENBURG; WIJNGAARD, 1994). Nevertheless, the U-line formulation still keeps simplifications on movement times between the line's sides. By considering any given layout with deterministic operator's movement times, an extension was lately developed by Sikora *et al.*

(2017a). Parallel lines and station paralleling variations also demand the relaxation of SH-8, and they are further discussed in Section 2.4.1.

The assembly line is designed for a single product (SH-9). To extend this assumption, mixed-model and multi-model features must be considered. This gives rise to sequencing and scheduling problems within the balancing one. Thomopoulos (1967) firstly presented a formulation for serial lines in which multiple products were taken into account. Later, Miltenburg (2002) relaxed SH-9 and applied multi-products concepts on U-shaped lines. More recent studies approached the balancing problem together with the sequencing one (SAWIK, 2012; ÖZTÜRK *et al.*, 2015; LOPES *et al.*, 2017; LOPES *et al.*, 2019; LOPES *et al.*, 2020b). A further extension of the combined problem is allowing the model to decide task assignments for balancing decisions and sequencing patters. Lopes *et al.* (2018a) integrated the problems in a cyclical formulation for unpaced lines and later included buffer allocation decisions (LOPES *et al.*, 2020a).

Lastly, parameters are deterministic (SH-10) and known, no stochastic considerations are made. It means that task processing times are determined parameters and do not vary at a probabilistic distribution. Relaxing this hypothesis implies on a line balancing that does not guarantee a fixed optimal cycle time, but just secure it with a certain probability (KOTTAS; LAU, 1973). A cost-oriented approach can be adopted in the said case. In mixed-model lines, the sequencing order can also be considered a stochastic factor (LOPES *et al.*, 2018; LOPES *et al.*, 2018b; LOPES *et al.*, 2020). When no sequencing order can be fixed but the probability of each product model is known, each product to enter the line can be seen as a random variable.

2.4 ASSEMBLY LINE DESIGN PROBLEM

Thus far, it has been declared that GALBPs are generalisations of SALBPs, since they consider relaxing one or any combination of the SALBP stated assumptions. Notwithstanding, GALBPs still omit some designing sub-problems, such as (i) selecting the equipment depending on the set of candidate solutions for each manufacturing operation, (ii) costs of dimensioning the production area, (iii) assigning heterogeneous workers or robots, and (iv) the layout option itself. Whenever these fixed and variable costs associated with the production system (facilities, technology, operation) are taken into account, SH-5 (all stations are equipped and manned to process any task) is relaxed and we have a further generalisation of the GALBP (BAYBARS, 1986), namely Assembly Line Design Problem (ALDP).

Although the literature on ALBPs is quite extensive, a gap between practical and

theoretical cases still exists, as may be noticed in the following surveys (BECKER; SCHOLL, 2006; BOYSEN *et al.*, 2008; BATTAÏA; DOLGUI, 2013). In real-world lines, flexible assembly systems are adopted and must be designed properly in order to reach the desired production rate at the minimum cost. These costs include facilities, programmable robots, workforce wages, and equipment selection. Therefore, global decisions made by companies necessitate and depend on the optimal solution for the line layout. These decisions come along with the objective of balancing assembly lines.

For industrial robotic lines, if managers desire to determine production system layouts beforehand, these designing decisions ought to be combined with cost minimisation procedures. Amen (2000), Amen (2006) provide a survey on heuristic procedures, model formulation and methods to solve cost-oriented ALBPs. Moreover, such designing decisions must take into account the possibility of parallel stations (Section 2.4.1), selecting equipment (Section 2.4.2), assigning multiple workers or robots to the same station (Section 2.4.3), and specific characteristics of resistance spot welding tasks (Section 2.5.1). These possibilities are studied in the Robotic Assembly Line Design (RALD) problem, which is proposed in Chapter 3 as part of this PhD thesis (MICHELS *et al.*, 2018b).

Rekiek *et al.* (2002) provide a comprehensive review for exact methods, heuristics, and meta-heuristics that cope with the ALDP.

2.4.1 Parallel Lines and Stations

Improving productivity in assembly lines is very important. It increases capacity and reduces cost. If the capacity of the line is insufficient, possible ways to increase it is to build parallel lines or stations. When an additional line is constructed or an identical station is placed in parallel with the original station in a serial line, the concept of parallel lines and stations is introduced in the assembly line. In this situation, each station has its cycle time doubled for its operations. Therefore, another particularity applied on line design problem is the inclusion of parallelism as an option for better balancing solutions. Tasks with longer processing times than the desirable cycle time or the requirement to improve system's efficiency are two circumstances in which these layouts are most useful.

Parallel lines and station paralleling extensions are approached by several authors. They research some potential advantages and drawbacks in paralleling, including equipment selection features, labour costs, positional constraints, and task assignment restrictions. Parallel lines and

stations also require more physical space, hence additional constraints should be considered. Lusa (2008) supplies a survey on parallelism applications in ALBPs, summarising the state of the art for the combined problem.

Regrading parallel lines, Gokçen *et al.* (2006) introduced the concept into ALBPs, generating the Parallel Assembly Line Balancing Problem (PALBP). Modern applications combine this layout with mixed-model lines (ÖZTÜRK *et al.*, 2015; LOPES *et al.*, 2017; LOPES *et al.*, 2019), giving rise to sequencing and scheduling problems within the PALBP.

When it comes to parallel stations, Bard (1989) firstly included the possibility of paralleling stations in order to diminish negative effects caused by dead times. Askin and Zhou (1997) used a heuristic procedure for the parallel station problem combined with a mixed-model line, Ege *et al.* (2009) promoted an extensive computational study regarding the effects of parallel station on ALBPs, and Tuncel and Topaloglu (2013) presented a real-world electronic product assembly line, in which the line's rebalancing is a constant problem and parallel stations are largely applied.

2.4.2 Equipment Selection

Tasks often require specialised equipment to be performed. Depending on the machinery, tool, or operator, task processing times may vary or, in extreme cases, be incompatible. If this type of situation occurs, an equipment selection problem comes along with it. In order to perform tasks faster, more efficient equipment is required, but it usually comes associated with higher costs. It follows that trade-offs are possible when deciding which is the most cost-effective group of equipment to be selected (BUKCHIN; TZUR, 2000). Additionally, a subset of tasks might need specific equipment to be performed.

The equipment selection problem is strongly related to the station paralleling one. Whenever a station is paralleled, so is its equipment (labour, tools, machinery), since both stations must be equally manned and equipped. A weighted approach was conducted by Bukchin and Rubinovitz (2003) in order to integrate station paralleling and equipment selection into an ALDP.

2.4.3 Worker Assignment

The inclusion of any worker assignment formulation normally requires the relaxation of SH-5 and, therefore, such problems are greatly related to ALDPs. The integration of worker assignment problems, however, have a separate category, which is known as the Assembly Line Worker Assignment and Balancing Problem (ALWABP). The ALWABP has been introduced by Miralles *et al.* (2008) and widely applied to sheltered centres for the disabled. Each worker has specific limitations and depending on the worker assigned to a station, task processing times might be higher or even infeasible there.

This idea can be expanded to any type of resources. The first generalisation of assignment restrictions is proposed by (SCHOLL *et al.*, 2010). A generic resource is constrained and must be balanced, either processing time, space, or operators should be summed as a finite resource.

Another variation of the worker assignment problem is presented by Sikora *et al.* (2017a). They approach travelling workers that might be able to perform tasks in more than one station, their movement times are measured and integrated into the worker's cycle time.

2.5 ROBOTIC ASSEMBLY LINES

Modern assembly lines rely on robotic workers performing automated tasks. For instance, several resistance spot welding (RSW) tasks (explained in Section 2.5.1) are commonly assigned to robots in automotive industries.

Early works on this field were introduced by Rubinovitz and Bukchin (1991), Rubinovitz and Bukchin (1993). They developed a method based on a branch-and-bound (B&B) algorithm that aimed to allocate robots into a (balanced) robotic assembly line. Moreover, the RALB problem introduced by Rubinovitz and Bukchin (1993) assumes different robots allocated in each station, creating a station dependence for task processing times. This model, however, does not analyse how workers (or robots, in the said case) behave in the production system when multi-manned stations, equipment selection, or parallel stations are considered.

Applicable variations of the GALBP might also be incorporated into the robotic models. This problem had its solution methods further studied in the past years (KIM; PARK, 1995; LEVITIN *et al.*, 2006; DAOUD *et al.*, 2014).

2.5.1 Resistance Spot Welding Tasks

The ALBP literature is tightened to the automotive industry, mainly to the final stage assembly. Figure 1, on page 14, illustrates how an automated welding line looks like. The arrangement of body-shop stages also consist of assembly lines that can be treated by ALBP formulations. The body-in-white stage transforms sheets of metal into the vehicle's body by using RSW procedures. This stage is composed of welding assembly lines that usually are highly automated (MICHALOS *et al.*, 2010).

Resistance spot welding tasks generally do not present precedence relations between welding spots. Nonetheless, some accessibility issues may occur when assembling new metal sheets together. These characteristics and their practical consequences are further explored in Chapter 3.

In a robotic welding assembly line, station paralleling possibilities might be included in the system in order to increase the line's efficiency, whereas equipment selection is primordial. In fact, some tasks can be performed with identical tools, but specific tools are required in order to execute a sub group of tasks. Multiple robots working in the same station can be used to shorten the line's length. Exceptionally due to the RSW tasks nature of not having precedence relations (LOPES *et al.*, 2017), robot task scheduling into each station can be neglected.

2.6 MULTI-OPERATED WORKSTATIONS

In industrial environments, the use of multi-operated workstations is intense. As aforementioned, assembly lines applied to automotive industry often produce large-size vehicles, such as cars and buses. In these lines, the SALBP's hypothesis of allowing only one worker in each station is not a practical limitation. As product size grows, it becomes possible to assign more than one worker to each station. These workers can simultaneously perform tasks in different sectors of the same product. This gives rise to natural extensions and more generalised versions of the SALBP: the Multi-manned Assembly Line Balancing Problem (MALBP) and the Two-sided Assembly Line Balancing Problem (TALBP), which are surveyed by Becker and Scholl (2006) along with a variety of practical extensions. The Assembly Line Balancing Problem with Variable Workplaces (VWALBP), introduced by Becker and Scholl (2009), can also be seen as a problem with multi-operated workstations.

For problems addressed in this PhD thesis, the most related concept is the use of

multi-manned stations, in which stations are occupied by multiple workers assigned to the same workstation (FATTAHI; ROSHANI, 2011; KELLEGÖZ, 2017; MICHELS *et al.*, 2019; MICHELS *et al.*, 2020). In Chapters 4 and 5, the development of this particular field of study is further discoursed in detail. This feature is commonly applied in the assemblage of large products, such as vehicles. The interference between workers and task scheduling in each station should be considered (BOYSEN *et al.*, 2008). The second and third parts of this PhD thesis focuses on the MALBP variant, this time taking into account task scheduling problems for each station; advantages of using the multi-manned configuration are associated to workforce and line length reduction or productivity gains, which are further exemplified in Chapter 4 and 5.

Two-sided assembly lines were firstly explored by Bartholdi (1993). The main difference between MALBPs and TALBPs is the flexibility on the quantity of workers and their positioning. TALBPs allow at most two operators, each of them on either the right or the left side, whereas in MALBPs the number of maximum workers depends on product's attributes, such as size, structure, and tasks' precedence relations. Another divergence is that TALBPs might have to deal with tasks that can be performed exclusively on the right or left side. Kim *et al.* (2000a) used a genetic algorithms to deal with the TALBP, while variants concerning mixed-model lines (ÖZCAN; TOKLU, 2009) and stochastic task times (ÖZCAN, 2010) were further developed. Finally, Guney and Ahiska (2014) decided the optimal automation level on an automotive industry operating with two-sided lines.

The last class of problems that cope with multi-operated workstations is the VWALBP (BECKER; SCHOLL, 2009). In this problem, working areas are minimised given a cycle time, while work-pieces are divided into mounting positions. Only one worker is able to assemble products in each position of a multi-manned station. A Mixed-Integer Linear Programming (MILP) model is generated including lower bounding techniques and a branch-and-bound algorithm is implemented to solve large-size instances. Naderi *et al.* (2019) recently used the idea of mounting positions to model and solve a realistic five-sided assembly line.

2.7 DEVELOPED WORK AND PROPOSED PROBLEMS

In this PhD thesis, the author firstly presents in Chapter 3 a mathematical model that minimises the designing cost of a robotic assembly line considering several realistic aspects found in the automotive industry: (i) different robots, (ii) their space and accessibility constraints, (iii) equipment selection for each of them, (iv) task assignment restrictions (incompatibility

and special precedence), (v) parallel stations, and (vi) unproductive time due to work-pieces movement between stations (dead time). The combination of these realistic aspects have the (SH-5)–(SH-8) relaxed and are not present in previous ALDP models, thus a new MILP model was proposed and solved. This work has been firstly introduced in Lopes *et al.* (2016) and later published in the *Computers & Industrial Engineering* under the title of "*The Robotic Assembly Line Design (RALD) problem: Model and case studies with practical extensions*" (MICHELS *et al.*, 2018b).

The following parts of this PhD thesis specifically copes with the MALBP aspects (Chapters 4 and 5). As RALD would simplify the multi-manned characteristic that arises when more than one worker is assigned to the same station, it may yield sub-estimated costs for the line design depending on problem parameters.

Chapter 4 gives a deeper definition of the MALBP in order to explain the problem, as well as a description of the presented MILP model and additional constraints to reduce the problem's search-space. Partial results of this work up to this point had been briefly presented in an abstract version Michels *et al.* (2018a). Moreover, this PhD thesis reviews the development and applications of Benders' decomposition and combinatorial cuts, used within the proposed solving framework. It also describes in detail the proposed algorithm. Computational results retrieved from this study are presented, discussed and concluding remarks are summarised. This complete work is published in the *European Journal of Operational Research* under the title of "*A Benders' decomposition algorithm with combinatorial cuts for the multi-manned assembly line balancing problem*" (MICHELS *et al.*, 2019).

Chapter 5 explores the type-2 MALBP. It lays out the main differences from the previous type-1 MALBP in order to explain the problem. A new MILP model with valid inequalities is presented. The development and applications of an enhanced Benders' decomposition algorithm with combinatorial cuts is also discussed. Experiments are conducted on a benchmark dataset (computational study) and a real-life assembly plant (case study). Results retrieved from these tests are presented, discussed and concluding remarks are summarised. This work is published in the *Operations Research Perspectives*, under the title of "*An exact method with decomposition techniques and combinatorial Benders' cuts for the type-2 multi-manned assembly line balancing problem*" (MICHELS *et al.*, 2020).

2.8 CONSIDERATIONS

This chapter presented the main features of the studied problems, namely RALD and MALBP. It described the common terminology, physical characteristics, and constraints of assembly lines. Additionally, balancing and scheduling problems found in the automotive industry were detailed alongside its operational constraints. The following Chapters 3, 4, and 5 are based on the content of full papers; hence, introduction, literature review, and problem statement sections of each one of these papers have been maintained to minimise changes in regards to original manuscripts.

3 THE ROBOTIC ASSEMBLY LINE DESIGN PROBLEM

This chapter contains a slightly modified version of the paper Michels *et al.* (2018b), which is entitled "*The Robotic Assembly Line Design (RALD) problem: Model and case studies with practical extensions*" and was published in the *Computers & Industrial Engineering*.

Section 3.1 presents a brief introduction and an extended version of the literature review on assembly line design problems and robotic workers. Section 3.2 defines the studied problem and the assumptions derived from spot welding tasks performed by robots. Naturally, there are some similarities with the general description of assembly lines given in Chapter 2; however, some definitions and assumptions will be explored in the specific context of a RALD problem. Section 3.3 describes the proposed MILP formulation for solving the general case of the problem and the practical case study. Section 3.4 contains the results and discussions of the developed approach, with computational and case studies. Final considerations of this chapter are presented in Section 3.5.

Abstract from Michels *et al.* (2018b): *"Spot welding assembly lines are widely present in the automotive manufacturing industry. The procedure of building the vehicle's body employs several robots equipped with spot welding tools. These robots and tools are a quite costly initial investment, requiring an efficient and conscious line design that meets product demands and minimises implementation expense at the same time. In this paper, the Robotic Assembly Line Design (RALD) problem is proposed and studied based on practical characteristics from an automotive company located in Brazil. A Mixed-Integer Liner Programming (MILP) formulation is developed allowing: (i) station paralleling, (ii) equipment selection, and (iii) multiples robots per workstation. The mathematical model aims at minimising the total cost at the desired production rate, which involves robots, tools and facilities. The proposed model considers dead time during a cycle, space constraints, task assignment restrictions, and parallelism possibilities. Dead time is an unproductive and fixed work-piece handling time included in the capacitated transporter robots' movement time. Computational experiments were performed in order to evidence the parameters' influence over the optimal line design solution. In addition, practical case studies were conducted with parameters collected from a real-world robotic welding assembly line located on the outskirts of Curitiba-PR (Brazil), reaching optimality. Compared to the strictly serial lines, the model led to great advantages by allowing parallel stations in the production system, making it possible to evaluate an expected trade-off between the production*

rate and the total cost; reductions of several hundred thousand dollars on the production layout cost can be achieved by the company, as indicated by the studied cases."

3.1 INTRODUCTION

Production systems used in the automotive industry are frequently based on assembly lines for regular operation. This variety of configuration has given rise to traditional assembly line balancing problems (ALBP), vastly approached in the literature. However, most part of the research focuses on achieving the best assignment of tasks among several stations arranged in a serial line considering many restricting assumptions, which not always describe and solve more realistic problems (BATTAÏA; DOLGUI, 2013).

Although the literature on ALBP is quite extensive, a gap between the practical and theoretical cases still exists (BECKER; SCHOLL, 2006; BOYSEN *et al.*, 2008; BATTAÏA; DOLGUI, 2013). In real-world lines, flexible assembly systems are adopted and must be properly designed in order to reach the desired production rate at the minimum cost, which includes facilities, programmable robots and equipment selection. Therefore, the global decisions made by the company necessitate and depend on the optimal solution for the line layout, which comes along with the balancing objective in an integrated problem.

Researches concerning ALBP are also strongly related to the automotive industry, mainly to the final stage assembly. Notwithstanding, the arrangement of body-shop stages also consists of assembly or manufacturing lines that can be treated by ALBP formulations (LOPES *et al.*, 2017). The body-in-white stage transforms sheets of metal into the vehicle's body by using welding procedures. This stage is composed of welding assembly lines that are usually highly automated (MICHALOS *et al.*, 2010). Welding procedures might present several spot welding tasks with similar characteristics. Formulations can take advantage of the high multiplicity of welding tasks and treat them as a group of similar tasks. Therefore, these similar tasks can be gathered together into a single task with a given number of copies of the same task, since their processing times are virtually identical (SIKORA *et al.*, 2017b). Moreover, welding tasks are found in the body-shop stage and these tasks fall into three main categories: geometry, stud, and finishing tasks. Geometry welding tasks assemble the metal sheet pieces together, stud welding tasks add screws on the metal sheets' surface, and finishing welding tasks are used to reinforce the vehicle's structure. Geometry and finishing tasks can be performed with identical tools, but a different tool is required in order to execute stud tasks.

According to Baybars (1986), these differences in describing realistic features divide the classification of ALBPs in two and make the problems fall into one of these categories: the Simple Assembly Line Balancing Problem (SALBP), or the General Assembly Line Balancing Problem (GALBP). In SALBPs, the system is only restricted by precedence relations and cycle time constraints, whereas GALBPs regard further specification, such as task incompatibility, space constraints, station paralleling, multiple and capable workers, equipment selection, or unproductive time, among others. An extensive review on SALBP is done by Scholl and Becker (2006), the same is done for GALBP by Becker and Scholl (2006), and a more recent survey on ALBPs is done by Battaïa and Dolgui (2013).

The solution methods for ALBPs are separated into exact and approximate approaches. The first one seeks the optimality, whilst the other includes heuristics and meta-heuristics procedures, which are intended to produce comparatively good results in a reduced computing time. Surveys on exact and heuristic methods are found in Scholl (1999), Scholl and Becker (2006) and Becker and Scholl (2006) for SALBP and GALBP, respectively. For meta-heuristics, some common improvement procedures applied in ALBPs are: tabu search (SUWANNARONGSRI; PUANGDOWNREONG, 2008), ant colony optimisation (SABUNCUOGLU *et al.*, 2009), simulated annealing (CAKIR *et al.*, 2011), genetic algorithms (SIKORA *et al.*, 2015), or local-search methods (SIKORA *et al.*, 2017c; MICHELS *et al.*, 2018).

Thus far, we can declare that GALBPs are generalisations of SALBPs, since they consider relaxing one or any combination of the SALBP stated assumptions. However, GALBPs still omit some designing sub-problems, such as selecting the equipment depending on the set of candidate solutions for each manufacturing operation, costs of dimensioning the production area, and the layout itself (BAYBARS, 1986). For instance, Bukchin and Rubinovitz (2003) show a case study in which different tasks can only be performed by different equipment, and distributing limited units of these tools is necessary. At the same time, parallel stations are required due to long task durations higher than the demanded cycle time, changing the line's configuration. Amen (2006) assumes each station to have a pre-specified investment. In automotive industry, these concepts can be extended to welding gun selection for particular tasks and capital costs of robots in an automated assembly line. Whenever these fixed and variable costs (facilities, technology, operation) are taken into account, we have a further generalisation of the GALBP, namely Assembly Line Design Problem (ALDP) (BAYBARS, 1986). For exact methods, heuristics and meta-heuristics that deal with the ALDP, Rekiek *et al.* (2002) provide a comprehensive

review. Nonetheless, as assembly line balancing and design problems are intimately linked, this nomenclature boundary is not always observed, and some design problems are named “balancing problems” by their authors. This fact is notably attested by the further explored literature.

Many of these ALBP extensions are explored individually. For instance, Rubinovitz and Bukchin (1991), Rubinovitz and Bukchin (1993) introduced the Robotic Assembly Line Balancing (RALB) problem, and a method based on branch and bound algorithm that aimed to minimise the number of workstations (type-1), while allocating robots into a (balanced) robotic assembly line. This model, however, does not analyse how workers (or robots, in the said case) behave in the production system when multiple robots, equipment selection, and parallel stations are considered. Cycle time minimisation variants (type-2) of the RALB problem were introduced by Levitin *et al.* (2006) and Gao *et al.* (2009). More recently, a multi-objective version of the type-2 RALB considering set-up and robot costs minimisation as secondary objectives has been presented (YOOSEFELAHI *et al.*, 2012).

Applicable variations of the GALBP might be incorporated in the robotic models. Multiple workers could be assigned to the same station (FATTAHI; ROSHANI, 2011; YAZGAN *et al.*, 2011), some workers might be able to perform tasks in more than one station and, hence, they would have to move between them (SIKORA *et al.*, 2017a), inclusion of workers with different capabilities or tools may be studied as in Araújo *et al.* (2015), that use disabled workers in the line, and ergonomic factors can be considered to decide workstation positioning on the line (BAYKASOGLU *et al.*, 2017). Multiple product design alternatives are investigated to influence balancing decisions: mixed-model assembly lines can be evaluated by several objective functions and require products to be sequenced, Lopes *et al.* (2020b) analyse and compare them considering buffer positions (if any) and product sequences as parameters, whereas Oesterle *et al.* (2017) incorporate such product design alternatives into balancing and equipment selection decisions and test their formulation with numerous multi-objective algorithms. Furthermore, tasks cannot always be assigned to any station (DECKRO, 1989), some set of specific tasks frequently must be allocated either at the same station (inclusion constraints), or are incompatible and must be placed at different stations (exclusion constraints) (SCHOLL *et al.*, 2010). Task incompatibility occurs when a pair of tasks cannot be assigned to the same station due to practical characteristics of the operation and, therefore, the precedence diagram is not sufficient to describe the precedence relations between tasks (PARK *et al.*, 1997). This particularity is also found in body-shop stages of automotive industries and is further explained in Section 3.2, by Figure 11.

Other incompatibility problems might also happen due to position or accessibility problems (ESSAFI *et al.*, 2010), fixed machinery (SCHOLL *et al.*, 2010), among others.

In parallel stations, each station has its effective cycle time multiplied by its parallelism degree, allowing more time for operations. Therefore, another particularity applied on line design problem is the inclusion of parallelism as an option for better balancing solutions when there are tasks with longer duration times than the desirable cycle time or the system's efficiency requires improvement. Lusa (2008) supplies a survey of parallelism applications in ALBPs, summarising the state of the art for the combined problem. Station and line paralleling extensions are approached by several authors, Askin and Zhou (1997) use a heuristic procedure for the mixed-model line with parallel stations, while Bard (1989) presents a dynamic programming algorithm which takes into account equipment and task costs, as well as unproductive time between cycles (dead time), which is mostly affiliated with transportation time between stations. The transportation time of work-pieces is attached to the movement of a conveyor or the time a robot takes to move a work-piece in (set-up) and out (tear-down) of a station, this usually is a fixed and unproductive handling time.

Some authors examine fixed and variable line configuration design costs in the planning stage, with decisions connected to the balancing problem, characterising such studies as ALDPs: Bukchin and Rubinovitz (2003), Ege *et al.* (2009), Dolgui *et al.* (2012), Yoosefelahi *et al.* (2012), and Tuncel and Topaloglu (2013) research potential advantages and drawbacks in station paralleling, including equipment selection, labour costs, positional constraints, and task assignment restrictions. Parallel stations require more space, hence additional constraints limiting the parallelism degree should also be considered. Both Kim *et al.* (2000b) and Guney and Ahiska (2014) consider a two-sided assembly line, the first one is dealt with using a genetic algorithms, whereas the other is solved by mixed-integer programming and is applied on an automotive industry in order to decide the optimal automation level. For industrial robotic lines, these designing decisions ought to be combined with cost minimisation procedures as to determine the production system layout. Amen (2000), Amen (2006) provide a survey on heuristic procedures, model formulations, and methods to solve cost-oriented problems. In order represent and solve practical problems, several herein mentioned real-world aspects must be taken into consideration simultaneously.

In this work, we develop a mathematical model that minimises the designing cost of an assembly line considering several realistic aspects found in the automotive industry: (i) robots,

(ii) their space and accessibility constraints, (iii) equipment selection for each of them, (iv) task assignment restrictions (incompatibility and special precedence), (v) parallel stations, and (vi) unproductive time due to work-pieces movement between stations (dead time). The combination of these realistic aspects are not present in previous ALDP models, thus a new model is herein proposed and solved. The chapter is organised as follows. In Section 3.2, the practical extensions of the problem are explained in order to introduce the Robotic Assembly Line Design (RALD) problem. Section 3.3 presents the Mixed-Integer Linear Programming (MILP) model developed for solving the general case of the problem and the practical case study. Section 3.4 analyses the case studies and the effects on the production system layout once the line designing cost ratios and unproductive times are altered. Lastly, in Section 3.5, concluding remarks are summarised and future research directions are suggested.

3.2 PROBLEM STATEMENT

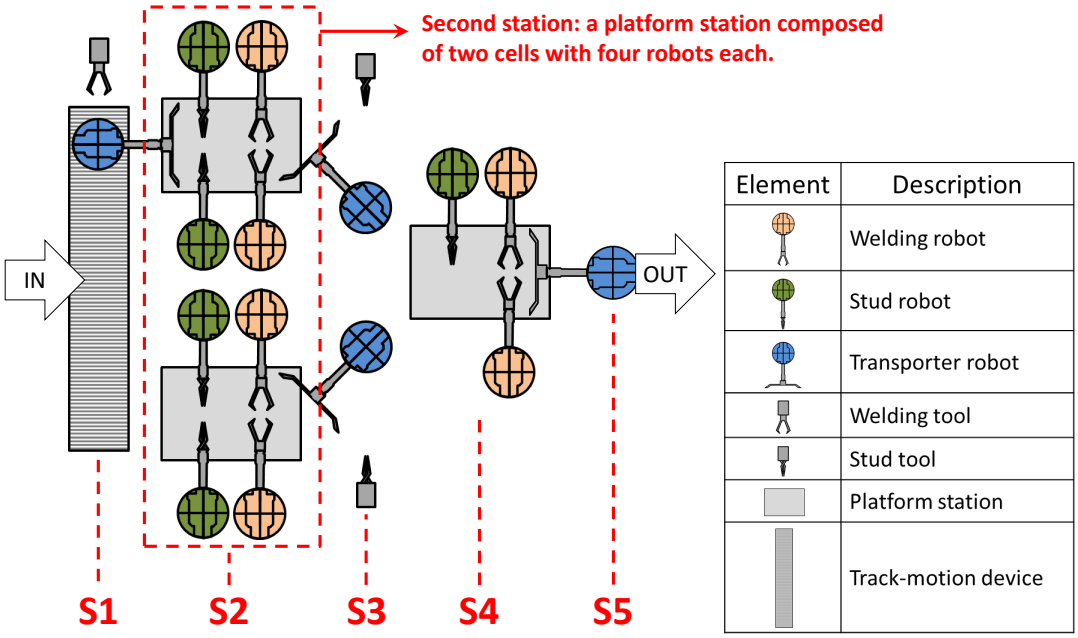
The Robotic Assembly Line Design (RALD) problem is proposed and studied in this chapter. We aim at the optimisation of an assembly line's layout. The decisions concern the length of the line along with the number and type of robots assigned to the line. Tasks requiring different types of tools are balanced within the line, while the transport system and parallel stations are also layout decisions of the problem. In order to ease the understanding of the problem's features, Figure 6 will be used as a visual support.

We consider that a robotic assembly line is composed of platform stations and transporter robots to displace work-pieces between these platforms, as illustrated by Figure 6. Multiple robots may be assigned to each platform station. By defining each transporter or platform as a station, and assuming the line starts and finishes with a transporter robot, the configuration results in a line with an odd number of stations in an alternating pattern of platform-fixed and transporter robots. More examples can be found in Figures 13, 14, 15, and 16 of Section 3.4.

In this example of Figure 6, the first transporter robot is on a track-motion device (S1) and has a welding tool placed on its side. The second station (S2) is a parallel platform station composed of two cells with four robots each. The third station (S3) is also paralleled, contains two cells of transporter robots, and each robot has a stud tool placed on its side. The forth station (S4) has only one cell and holds three robots. The fifth and last station (S5) is a single transporter cell without any task performing tool placed sideways.

The line allows parallelism of both platform and transporter stations. However, when a

Figure 6 – Example of elements in the proposed RALD problem are shown: parallelism in platform and transporter stations, multiple platform and transporter robots holding different tools or placed sideways, and transporter robots on track-motions.



Source: Michels *et al.* (2018b).

non-paralleled transporter is adjacent to a paralleled platform station, a track-motion device is required. Notice that the track-motion device applied on station S1 is necessary for its transporter robot to reach both cells of station S2.

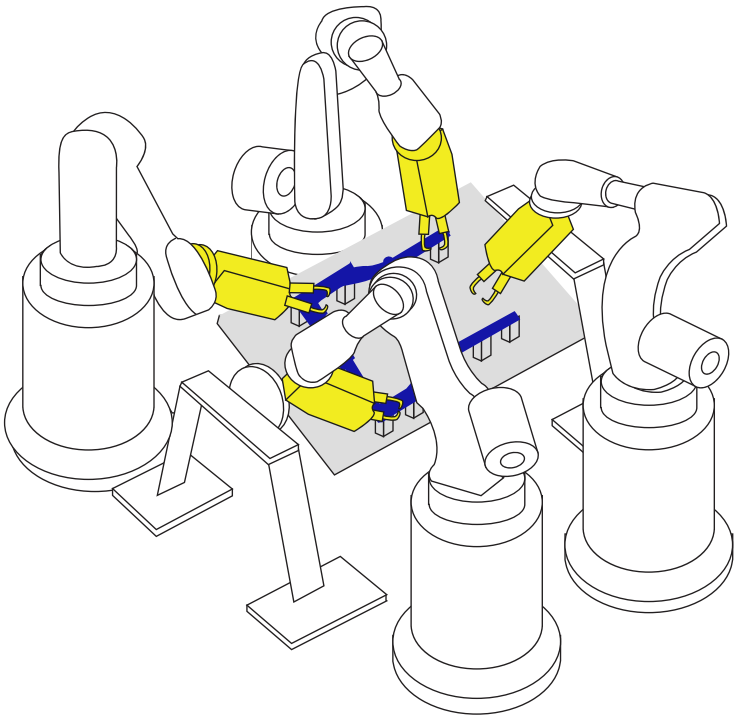
The robots are assigned to platform or transporter stations composed of one cell (single stations) or two identical parallel cells (double stations). The transportation of work-pieces between the stations is done by manipulator robots. Commonly, these movement times (dead times) are neglected in the problem formulation, a simplification that may lead to unreliable solutions. An advantage of using robotic arms as transporters is that they can also perform tasks on work-pieces during a cycle time between their loading and unloading operations.

Platform robots holding welding tools are dedicated exclusively to perform assembly tasks, while transporter robots are mainly used for moving products in and out of the stations, but they also can use the remaining time of their cycle to perform tasks as long as they have a static tool placed on the line's sideways. This condition can be observed in stations S1 and S3 of Figure 6. These transporter robots that are capable of performing tasks are named capacitated transporter robots. Such robots are under the effect of an increased time penalisation on task time duration, since they take longer to perform the same tasks robots in platform stations do. This increment in the task time duration happens because, whereas in platform stations the

work-pieces are steady for the robot to access the spot welding points, in transporter stations the robots have to manipulate the entire work-piece in order to make the points accessible to the tool. For many industry segments, tools are usually lighter and smaller than the produced work-pieces.

Figure 7 is a picture of platform robots assembling a work-piece (highlighted in blue) by performing welding tasks. The welding guns (highlighted in yellow) are held by the robots and the product stays fixed in the station. As the movement between welding spots is small, so is the processing time of the tasks.

Figure 7 – Platform robots performing welding tasks at the same time on the same blue work-piece. The yellow welding guns that each robot holds are highlighted in the picture.

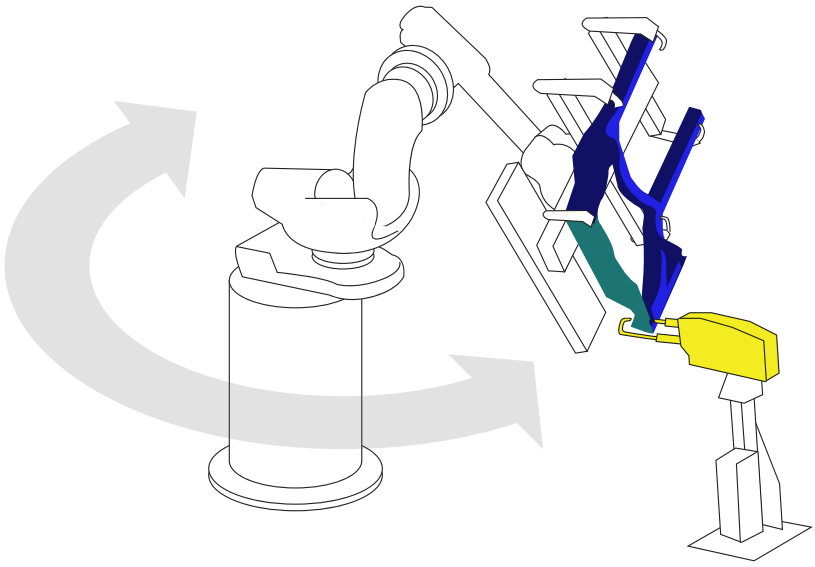


Source: Michels *et al.* (2018b).

Figure 8, on the other hand, is a picture of a transporter robot that uses a static welding tool. The yellow welding gun, in this case, is placed in the line's sideways and the robot moves the entire blue work-piece in order to make the welding spots accessible to the welding gun and perform the required tasks in a diminished rate. Platforms and capacitated transporter robots can be very similar. In our example, the same robot model can be employed both in platform and transport stations, the difference lays on the tools attached to them.

Generally, robotic arms are not long enough to reach both sides of a parallel station (at least for large work-pieces, such as vehicle parts, for instance). These transporter robots might be placed on track-motion devices in order to allow them to reach adjacent paralleled stations

Figure 8 – Transporter robot holding the entire work-piece and manipulating it to the welding gun placed on the line's sideways. The work-piece that the robot holds and the static yellow welding gun are highlighted in the picture.



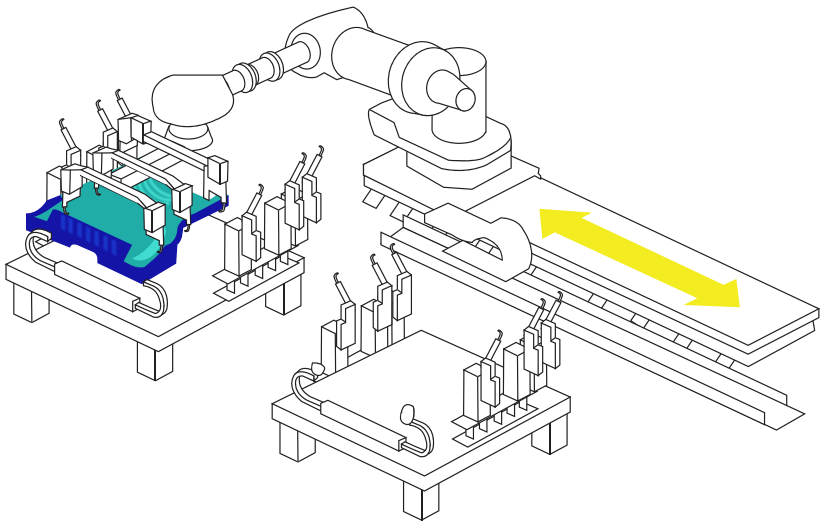
Source: Michels *et al.* (2018b).

and avoid unnecessary transporter station paralleling. Although the device allows more reach to the robot, an additional unproductive time penalty tied to the track-motion device movement should be accounted on the robot's cycle time. Figure 9 portrays a transporter robot placed on a track-motion device set to unload a platform station. The arrow placed on the track-motion device shows the movement that it permits. Figure 6 depicts this alternated pattern between platform and transport stations, as well as equipment disposition and several other characteristics of the RALD problem from a simplified top side view. Parallel station advantages and spot welding processes are explained separately in this section.

Firstly formulated by Rubinovitz and Bukchin (1991) and later dealt with a genetic algorithm (GA) by Levitin *et al.* (2006), the robotic assembly line balancing (RALB) problem is incorporated into the problem we have at hand. Some general assumptions can be kept from their work since the balancing core characteristic for the single product problem is present in our formulation:

1. The cycle time to meet the demand is known.
2. The assembly tasks precedence diagram is known.
3. There are no precedence relations within a station.
4. The duration of a task is deterministic and cannot be subdivided.

Figure 9 – Transporter robot on a track-motion device unloading a work-piece from a parallel platform station.



Source: Michels *et al.* (2018b).

5. Robots and equipment are available at any quantity.

The specific problem has some differences. They are highlighted in bold in the following assumptions:

6. The duration of a task depends on **which robot and tool is assigned** to perform it.
7. **Parallel stations** are allowed.
8. For the general case, any task can be performed at any station when the precedence relations and **equipment requirements** are attended.
9. **Multiple robots** may be assigned to each station on the line not considering task scheduling within stations.
10. Transportation time for loading (set-up) and unloading (tear-down) are considered. Therefore, **dead time is considered** (BARD, 1989).
11. The goal is to **minimise design cost**, both robots and equipment have their prices as parameters.

These highlighted aspects distinguish the classical RALB problem from our RALD problem: lines are not strictly serial, i.e. either platform or transporter workstations can be doubled (see second and third stations in Figure 6) in order to improve efficiency at a reduced

cost, unproductive transportation time is considered in the mathematical formulation, and multiple robots holding different tools are allowed at the same station (see second and forth stations in Figure 6).

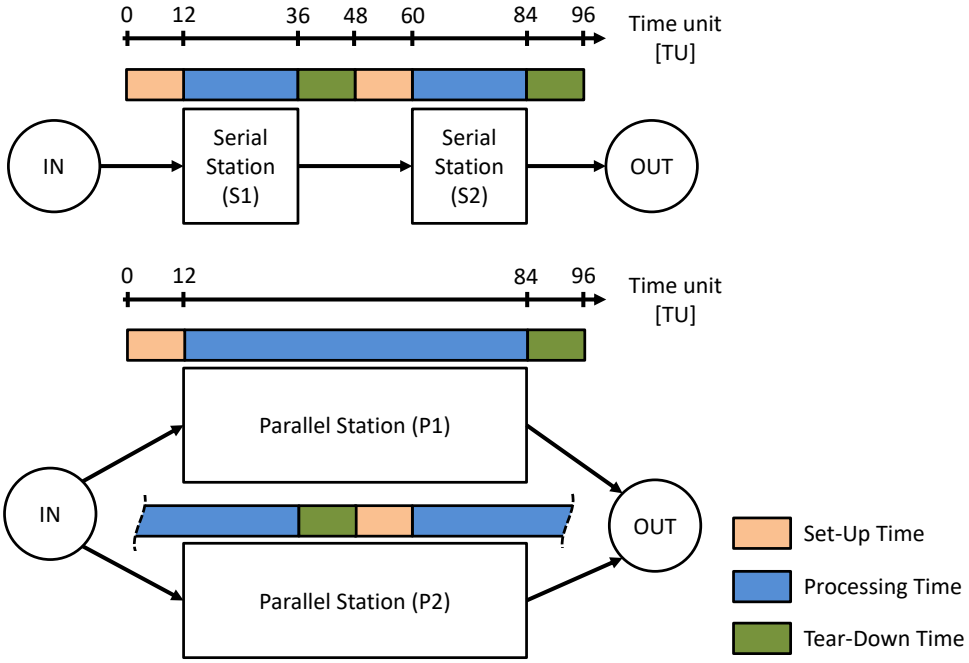
Station paralleling might lead to some advantages over single lines. The cycle time increase at the doubled station is one of them, mainly when the dead time is considered, since the relative importance of movement times (set-up and tear-down) is reduced. This effect is depicted in Figure 10, showing how parallel stations affect the line efficiency positively.

In Figure 10, cycle times of two serial and parallel stations are presented on a schematic diagram. In the first case (S1 and S2), for a given cycle time of 48 time units, set-up and tear-down times of 12 time units each, the processing time (useful time) represents only 50% of the cycle time (24 time units represented by the larger block placed on its top) and set-up and tear-down (dead time) the other 50% (12 time units each represented by the smaller blocks). When the station is doubled, one work-piece should be delivered per station every two cycles, in an alternated fashion. Each copy of it (P1 and P2) benefits from paralleling and increases its useful time to 75% (72 time units) of their doubled cycle time (96 time units). Therefore, there is more available time to be dedicated to task performing activities, an improvement of 50% in the useful time. The work-piece handling time (dead time) is the same for both configurations when loading (set-up) or unloading (tear-down) stations. However, the relative importance of this manipulation time is reduced because such work-piece handling process is conducted fewer times. The efficiency gain depends on which tasks were allocated to the serial stations and which could be allocated to an equivalent paralleled one.

Moreover, secondary advantages of paralleling stations are the improvement in productivity as a consequence of better balancing (BOYSEN *et al.*, 2007) and the reduction on failure sensitivity, albeit the reduced production rate (REKIEK *et al.*, 2002). These reasons make parallel stations a potentially profitable feature to be allowed into the line for practical applications. Nonetheless, there are some drawbacks in the practical use: doubled stations might require greater investments on equipment and higher operational costs. Not only the costs represent a trade-off, also larger space is required for the double stations, robots and equipment, which forces limits on the number of robots per workstation and paralleling degree.

Even disregarding the aforementioned drawbacks, paralleling stations indefinitely is not always possible depending on which tasks one is performing. For instance, the automotive manufacture widely uses the Resistance Spot Welding (RSW) technique in order to perform

Figure 10 – Schematic comparative between serial and parallel stations. The benefits of paralleling a station increase as the dead time represents a higher proportion of the cycle time.



Source: Michels *et al.* (2018b).

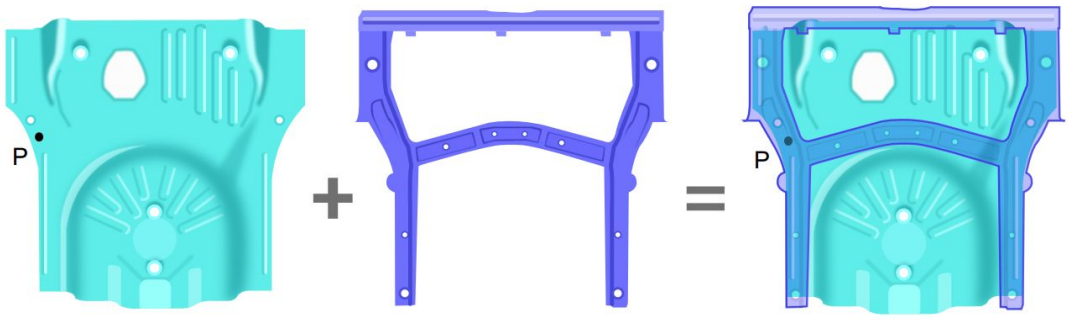
welding tasks in the body-in-white stage. This process unites metal sheets by using welding guns, which, in turn, require accessibility on both sides of the piece and, therefore, multiple types of this tool may be necessary as the welding steps proceed. In addition, metal sheet joining tasks must respect geometric tolerances, demanding external actuators to bind the metal sheets to be united in the proper position. Such tasks are called **geometry** tasks. Due to precision conditions, these geometry tasks must be performed at platform stations and the stations in which they are processed **must not be paralleled** for quality control.

Furthermore, the RSW technique is also used for reinforcement welding and screw adding points, namely **finishing** and **stud** tasks, respectively. Differently from geometry tasks, these ones do not require actuators to assure geometric tolerances and, theoretically, there are no precedence relations among any welding point. However, these spot welding points might become inaccessible after geometry tasks are performed, since the newly added sheets block the access of inner layers. These blockages can be seem as station-wise accessibility windows that can be represented as a special precedence diagram (see Figure 12 on page 60). Moreover, this welding assembly line property creates an incompatibility restriction between piece joining tasks (geometry tasks) and reinforcement tasks (finishing and stud tasks): all reinforcement tasks must be completed a station before any successor geometry task is allocated, since piece joining must

be the first operation performed at a station and reinforcement tasks would not be accessible after joining parts.

Figure 11 exemplifies why task incompatibility exists. Two car parts have to be joined, which requires geometry tasks. This operation must be the first to be performed at a station due to the use of external actuators. Yet once it is completed, some reinforcing welding procedures (finishing and stud tasks on the base pieces) become inaccessible for the robots in the assembled set. The region that becomes inaccessible is denoted by the area in which an overlap is verified between the base pieces. For instance, consider a spot welding point P that requires a finishing task and belongs to the first base piece, as illustrated by Figure 11. As stated before, welding procedures need access to both sides of the metal sheet to be performed. If the required finishing task is not performed on P before joining the base pieces, this point would be inaccessible afterwards, due to an inner layer blockage.

Figure 11 – Two car parts are assembled by platform robots performing geometry tasks. After the assemblage, the welding point P becomes inaccessible to further reinforcing welding procedures, therefore creating an incompatibility restriction between specific tasks.



Source: Michels *et al.* (2018b).

The assemblage of succeeding metal sheets is the only reason for the precedence relations between tasks. Since incompatible tasks must be performed at different stations (and before being inaccessible), it is possible to use multiples robots not considering the task scheduling within a station by the automotive industry. This is further discussed in Section 3.3.

Therefore, after the presented considerations, the configuration of an entire robotic line can be defined, conceiving the Robotic Assembly Line Design (RALD) Problem: how many robots per platform and transporter stations should be installed, how many tools of each type to use, which tasks are assigned to each robot considering equipment availability, and which stations would need to be paralleled in order to meet the demand at minimum cost. These decisions must all be made simultaneously to assure the optimal solution. The summary of elements in the optimisation model is shown in Figure 6 and are hereafter detailed as:

- **Parallelism possibilities:** both platform and transporter stations are allowed to be single or double stations;
- **Platform and transporter robots:** multiple robots per platform cell are allowed, and both platform and transporter robots are capable of performing tasks as long as they have a tool assigned to them;
- **Equipment selection:** different types of tools may be assigned to platform and transporter robots, even at the same station;
- **Track-motion device possibility:** transporter robots might be placed on track-motion devices, this feature is required for single transporters adjacent to double platforms due to the size of the products to be assembled.

Ultimately, if one knows the productivity rate to meet the demand (desired cycle time of the line), the best solution will be the one that accomplishes this need and satisfies the precedence and space constraints at the lowest cost.

3.3 ROBOTIC ASSEMBLY LINE DESIGN (RALD) MODEL

This section contains a Mixed-Integer Linear Programming (MILP) formulation for the Robotic Assembly Line Design (RALD) considering the problem definition and its characteristics described in Section 3.2. In order to ease the model's understanding, a concept based on the initial letter orientation will be employed for the parameters, sets, and variables definition as follows: all parameters and sets are written with an initial capital letter, an initial "*b*" indicates a binary variable (domain in $\{0,1\}$) and an initial "*v*" indicates a non-negative continuous (domain in $\mathbb{R}_+$) or integer variable (domain in $\mathbb{Z}_+$).

Table 2 contains the applied terminology, describing the parameters and the sets used in the formulation. The parameters are empirically collected based on industrial conditions, such as available physical space and suppliers' prices. Note that the maximum number of stations (NS) must always be an odd number due to: (i) work-piece initial handling and final manipulation out of the line, and (ii) the transporter-platform sequential stations characteristic of the problem, which make transporter stations to be indexed with odd numbers and platform stations with even numbers. Figure 6 on page 42 illustrates a line with an odd number of stations planned in an

alternated pattern. The variables are detailed in Table 3, they are created by the model depending on the sets.

Table 2 – Terminology: names of parameters and sets, their meaning, and [dimensional units].

Parameter	Meaning
NT	Number of tasks
NS	Maximum number of stations (an odd number, $NS \geq 3$)
$Nmax$	Maximum number of robots per cell
CT	Cycle time [time units]
DT	Dead time [time units]
TM	Time penalisation for track-motion [time units]
$D_{t,e}$	Duration [time units] of task t performed by equipment e
N_t	Number of copies of task t
$PCost$	Platform cost [$]
$TMCost$	Track-motion cost [$]
$RTCost_e$	Transporter robot cost [$] holding equipment e
$RPCost_e$	Platform robot cost [$] holding equipment e
Set	**Meaning**
T	Set of tasks t
S	Set of stations s
St	Set of transporter stations s, odd stations in S
Sp	Set of platform stations s, even stations in S
TS	Set of feasible Task-Station elements
SE	Set of feasible Station-Equipment elements
TSE	Set of feasible Task-Station-Equipment elements
$Prec$	Set of precedence relations between two tasks t_i and t_j: (t_i,t_j)
SS	Set of tasks that require single stations
Inc	Set of incompatible tasks t_i and t_j: (t_i,t_j)

Source: Michels *et al.* (2018b).

Table 3 – Terminology: definition of the model's variables.

Variable	Set	Domain	Meaning
$vTd_{t,s,e}$	$(t,s,e) \in TSE$	$\mathbb{Z}_+$	**T**ask **d**esignation: set to the number of copies of task t assigned at station s using equipment e
$bTe_{t,s}$	$t \in T, s \in S$	$\{0,1\}$	**T**ask **e**nding: set to 1 if all the copies of task t are finished up to station s
$bTo_{t,s}$	$(t,s) \in TS$	$\{0,1\}$	**T**ask **o**ccurrence: set to 1 if any copy of task t is performed at station s
bSo_s	$s \in S$	$\{0,1\}$	**S**tation **o**pened: set to 1 if station s needs to be used
bSd_s	$s \in S$	$\{0,1\}$	**S**tation **d**oubled: set to 1 if station s needs to be parallel
bTM_s	$s \in St$	$\{0,1\}$	Transporter station with **t**rack-**m**otion: set to 1 if the robot in transporter station s is on a track-motion device
$vNR_{s,e}$	$(s,e) \in SE$	$\mathbb{Z}_+$	**N**umber of **r**obots per cell in the station s holding equipment e
$vTNR_{s,e}$	$(s,e) \in SE$	$\mathbb{Z}_+$	**T**otal **n**umber of **r**obots per station s holding equipment e
$vUT_{s,e}$	$(s,e) \in SE$	$\mathbb{R}_+$	**U**seful **t**ime at the station s using equipment e [time units]

Source: Michels *et al.* (2018b).

The problem's objective function is to minimise the purchase cost of the line. As it is

described in Expression 1, the line's cost is composed of the cost of the platform and transporter robots along with their tools, platforms, and track-motions.

$$\text{Minimise: } \underbrace{\sum_{\substack{(s,e)\in SE \\ s\in Sp}} RPCost_e \cdot vTNR_{s,e}}_{\text{platform robots' cost}} + \underbrace{\sum_{\substack{(s,e)\in SE \\ s\in St}} RTCost_e \cdot vTNR_{s,e}}_{\text{transporter robots' cost}} + \\ + \underbrace{PCost \cdot \sum_{s\in Sp}(bSo_s + bSd_s)}_{\text{platform cost}} + \underbrace{TMCost \cdot \sum_{s\in St} bTM_s}_{\text{track-motion cost}} \tag{1}$$

The layout planning depends on balancing tasks among workstations. For each workstation, the number of robots, station paralleling, and the presence of track-motion devices delimit the available time for operations to be performed. The inequalities for the balancing core of the model (Inequalities 2 to 5) are based on the formulation for the high dimensionality (Integer) SALBP, presented by Sikora *et al.* (2017b) and applied on real-world instances from an automotive body welding assembly line. In this alternative formulation, decision variables for each existent type of task are defined and replicas of a task are treated in a single integer variable. Therefore, instead of creating binary variables to determine whether a task is assigned to a station or not, the integer based formulation relies on integer variables that decide how many copies of each type of task are assigned to each station. Assembly lines with high multiplicity of identical tasks (e.g. resistance spot welding tasks) may be modelled using fewer variables than traditional binary based formulations.

Equation 2 is the occurrence restriction. While binary based formulations allocate every task individually, Equation 2 states that the sum of all allocations of task t assigned to stations must be equal to its number of copies N_t. The precedence restriction is given by Inequality 3. Each task can only be assigned to a station (by task designation variable vTd) if all of its predecessors have already been completed (measured by task ending variable bTe) in or before a station s. The link between vTd and bTe variables for the same task is given by Inequalities 4 and 5. Inequality 4 assures bTe can only assume 1 if all N_t copies of the task are already performed up to station s, hence allowing followers to be assigned. Complementary, Inequality 5 forces $bTe = 1$ when the task is finished. This last restriction can be seen as a cut. Although it is not necessarily mandatory to formulate the problem correctly, it can help to tighten the formulation.

$$\sum_{(t,s,e)\in TSE} vTd_{t,s,e} = N_t \qquad \forall\, t \in T \tag{2}$$

$$bTe_{t_i,s} \cdot N_{t_j} \geq vTd_{t_j,s,e} \qquad \forall\,(t_i,t_j)\ \in\ Prec,\ (t_j,s,e)\ \in\ TSE \tag{3}$$

$$bTe_{t,s} \leq \sum_{\substack{(t,sa,e)\in TSE \\ sa\leq s}} \frac{vTd_{t,sa,e}}{N_t} \qquad \forall\, t\ \in\ T,\ s\ \in\ S \tag{4}$$

$$bTe_{t,s} + N_t - 1 \geq \sum_{\substack{(t,sa,e)\in TSE \\ sa\leq s}} vTd_{t,sa,e} \qquad \forall\, t\ \in\ T,\ s\ \in\ S \tag{5}$$

Bukchin and Rubinovitz (2003) state that there is a diminishing return in paralleling stations. The first station in parallel improves the efficiency, whereas the contribution of additional parallel stations is quite small. Moreover, having many parallel station is only needed due to long task times, which do not happen in spot welding assembly lines (SIKORA *et al.*, 2017b). Besides, the cost of adding another copy of a robotic cell and the transporter accessibility to the stations would result in unaffordable or infeasible production layouts due to the size of vehicles. Therefore, we only consider possibilities of single or double stations.

The variable vUT is responsible for the measurement of the useful time used to perform tasks, and is calculated by summing the performed tasks (Equation 6). The available time depends on whether the station is open, single or double, and the number of robots. Inequality 7 presents a limit for the variable vUT based on the available time. The bold terms are variables. Note that the equation is not linear: the useful time depends on the product of three variables, posing a linearisation challenge.

$$vUT_{s,e} = \sum_{(t,s,e)\in TSE} vTd_{t,s,e} \cdot D_{t,e} \qquad \forall\,(s,e)\ \in\ SE \tag{6}$$

$$vUT_{s,e} \leq \boldsymbol{bSo}_s \cdot [(1 + \boldsymbol{bSd}_s) \cdot CT - DT] \cdot \boldsymbol{vNR}_{s,e} \qquad \forall\,(s,e)\ \in\ SE \tag{7}$$

This non-linear expression can be decomposed in the linear expressions 8, 9, and 11. Inequality 8 is used to determine whether the station is open: if bSo is 0, there is no useful time in the station; otherwise, the value $(2 \cdot CT - DT) \cdot Nmax$ is an upper bound for the useful time in a station. If the station is open and single, Inequality 9 is dominant. A non-doubled station ($bSd = 0$) results in restricting the useful time to $(CT - DT) \cdot vNR$. Inequality 10 is only applied to the transport stations (St), also considering a time penalisation for the use of track-motion devices. Once a robot using track-motion has to move between two workstations,

there is less available time for the performance of tasks. Finally, Inequality 11 restricts the useful time when a station is doubled.

$$vUT_{s,e} \leq \boldsymbol{bSo}_s \cdot (2 \cdot CT - DT) \cdot Nmax \qquad \forall\,(s,e)\ \in\ SE \tag{8}$$

$$vUT_{s,e} \leq (CT - DT) \cdot \boldsymbol{vNR}_{s,e} + \boldsymbol{bSd}_s \cdot Nmax \cdot CT \qquad \forall\,(s,e)\ \in\ SE \tag{9}$$

$$\sum_{s,e \in SE} vUT_{s,e} \leq \sum_{s,e \in SE} (CT - DT) \cdot \boldsymbol{vNR}_{s,e} + \boldsymbol{bSd}_s \cdot Nmax \cdot CT - \boldsymbol{bTM}_s \cdot TM \qquad \forall\, s\ \in\ St \tag{10}$$

$$vUT_{s,e} \leq (2 \cdot CT - DT) \cdot \boldsymbol{vNR}_{s,e} \qquad \forall\,(s,e)\ \in\ SE \tag{11}$$

Due to space and accessibility constraints, the number of robots per cell (vNR) must be limited (Inequality 12). The total number of robots per station ($vTNR$) depends on whether the station is doubled. For example, in Figure 6, the second station contains four robots per cell ($vNR = 4$), however, as that station has been doubled, the total number of robots in the station is eight ($vTNR = 8$). This could be stated as a non-linear equation ($vTNR_{s,e} = vNR_{s,e} \cdot (1 + bSd_s)$), but, in order to keep a linear formulation, a decomposition is required. Hence, Inequality 13 measures the number of robots for single stations, while Inequality 14 is active for double stations. Notice that this multi-robots aspect is only possible because there are no precedence relations within a station. Otherwise, task scheduling would be necessary in order to assure feasible answers. Nonetheless, if one sets the maximum number of robots per cell to one ($Nmax = 1$) in Inequality 12, the model is still valid for problems with a single robot per station. It is also important to notice that $vTNR$ is minimised in the objective function and, therefore, the variable $vTNR$ is set to receive the value of the number of robots per cell (vNR) depending on the parallelism applied to the station in Inequalities 13 and 14.

$$\sum_{(s,e) \in SE} vNR_{s,e} \leq Nmax \qquad \forall\, s\ \in\ Sp \tag{12}$$

$$vTNR_{s,e} \geq vNR_{s,e} \qquad \forall\,(s,e)\ \in\ SE \tag{13}$$

$$vTNR_{s,e} \geq 2 \cdot vNR_{s,e} - 2 \cdot Nmax \cdot (1 - bSd_s) \qquad \forall\,(s,e)\ \in\ SE \tag{14}$$

The next restrictions control the shape and flow of the line. Firstly, adjacent stations can only be opened if a previous one has already been opened (Inequality 15). A parallel station has two identical cells (same number of robots and equipment) performing the same tasks. In order to duplicate a cell, it firstly needs to exist and be operative. Inequality 16 assures that a station can only be doubled if it is open. As the product flows across the line using transporter robots, it is necessary to have at least one of them at the starting point and after all platform stations in an alternated manner (see Figure 6). The transport cells are considered to contain only one robot and must both start and finish the assembly line. These restrictions are represented by Equality 17 for the first station and Equality 18 for the remainder stations. Note that Equality 17 is only applied to $s = 1$ and Equality 18 for $s \in St$, such that $s > 1$.

$$bSo_s \leq bSo_{s-1} \qquad \forall\ s\ \in\ S \mid s > 1 \tag{15}$$

$$bSd_s \leq bSo_s \qquad \forall\ s\ \in\ S \tag{16}$$

$$\sum_{(s,e)\in SE} vNR_{s,e} = 1 \qquad \forall\ s\ \in\ S \mid s = 1 \tag{17}$$

$$\sum_{s,e\in SE} vNR_{s,e} = bSo_{s-1} \qquad \forall\ s\ \in\ St \mid s > 1 \tag{18}$$

Furthermore, either when a work-piece has to be transported from a single transporter robot into a doubled platform station or vice-versa, the transporter robot requires to be on a track-motion device (see Figure 9), unless this transporter station has also been paralleled (Inequalities 19 and 20). In Figure 6, S2 is a platform station that has been doubled, consequently, the transporter robot before it (S1) had to be placed on a track-motion device in order to make the robot reach both platform cells and deposit work-pieces correctly. Alternatively, one could duplicate a transportation cell instead, as it happened to S3, in which each transporter robot is responsible for unloading work-pieces from different cells in the previous parallel station. Inequality 19 controls the use of a track-motion device when moving work-pieces from single to parallel stations and Inequality 20 the other way around. Note that both restrictions are valid for $s \in St$ and variables bTM_s must assume 1 when the respective transporter station is not double, i.e. $bSd_s = 0$.

$$bTM_s \geq (1 - bSd_s) + bSd_{s+1} - 1 \qquad \forall\ s\ \in\ St \mid s < NS \tag{19}$$

$$bTM_s \geq (1 - bSd_s) + bSd_{s-1} - 1 \qquad \forall\, s \in St \mid s > 1 \tag{20}$$

Up to this point, the model is sufficient to describe the basic Robotic Assembly Line Design presented in Section 3.2. There are, however, some extra practical restrictions in the case study of Section 3.4 that require more expressions. As it is stated in Section 3.2, some pair of tasks may require a single station, and some tasks cannot be performed in the same station.

The modelling of extra restrictions requires an auxiliary binary variable (bTo) that controls whether any copy of task t is performed at a station s. The link between bTo and the number of copies of tasks allocated to a station (vTd) is given by Inequalities 21 and 22.

$$bTo_{t,s} \geq \sum_{(t,s,e) \in TSE} \frac{vTd_{t,s,e}}{N_t} \qquad \forall\, (t,s) \in TS \tag{21}$$

$$bTo_{t,s} \leq \sum_{(t,s,e) \in TSE} vTd_{t,s,e} \qquad \forall\, (t,s) \in TS \tag{22}$$

Due to technological restrictions (quality in precision, process control constraints), geometry welding tasks are required to be performed on single platform stations, and all the precedent tasks must be completed one station before the geometry tasks start. Therefore, Inequality 23 is needed to assure that if a geometry task is performed in station s, the station cannot be doubled ($bSd = 0$). The necessity of finishing all precedence tasks before a geometry task can be modelled with a precedence relation (Inequality 3) added to the effect of an exclusion constraint due to incompatibility (Inequality 24).

$$1 - bTo_{t,s} \geq bSd_s \qquad \forall\, t \in SS,\ (t,s) \in TS \tag{23}$$

$$bTo_{t_i,s} + bTo_{t_j,s} \leq 1 \qquad \forall\, (t_i,t_j) \in Inc,\ (t_i,s) \in TS,\ (t_j,s) \in TS \tag{24}$$

3.4 RESULTS

Two datasets were developed based on real-world data and computational tests were performed in order to examine the influence of several model's parameters and validate the model. This first case study also seeks to evaluate the influences of model's parameters in computational difficulty. The complete mathematical formulation, including extensions (Constraints 21 and 22) and extra restrictions (Constraints 23 and 24), was applied on them. The first dataset is composed

of basic robots and tools, whilst the second one was elaborated with the same data in an enlarged equipment pool. These results are presented in Section 3.4.1.

Moreover, the model is tested on real-world data that has been collected from an automotive industry located on the outskirts of Curtiba-PR (Brazil) and converted into practical instances in order to analyse three case studies for different vehicle models produced in the company. Each vehicle model requires different amount of copies of each task and the duration of each copy may also be different depending on the vehicle model. The last model is the most complex one, presenting more tasks to be performed. The production rate to meet the demand is known and the assembly welding line ought to be designed aiming to achieve the desired cycle time at the lowest cost. The obtained results (Table 8) for the optimal line and the line as it currently is implemented are compared and discussed by Table 9 in Section 3.4.2, suggesting a potential economy of 5.9%.

To all instances, a 64 bit Intel™ i7 CPU (2.9 GHz) with 8 GB of RAM was employed using eight threads and the IBM ILOG CPLEX Optimization Studio 12.6. Optimal solutions were found for all instances in the first set (Section 3.4.1, Table 4) of the computational experiments and practical cases (Section 3.4.2, Table 8) within 3600 seconds, not exceeding the solving time limit. For the enlarged set, 18 out of 32 instances were solved within the time limit (Section 3.4.1, Table 5).

3.4.1 Parameters' Influence Computational Study

It has been shown in Figure 10 that the dead time can be diluted between parallel stations. As the relative importance of the dead time (DT) in regard of the cycle time (CT) increases, it is expected that the line design will converge towards solutions with more parallel stations, so as to reduce the negative effects of unproductive movements and product loading. To observe this behaviour, computational tests were performed varying the DT from 0 to 70% of the CT. Larger values of DT were neglected for functional reasons: no line would operate with such inefficiency and CPU processing time is much higher when the number of maximum stations is increased. Consequences of this fluctuation on DT can be detected on the number of robots, use of track-motions, and the final cost.

Another computational experiment has been conducted in order to analyse the effects of changing cost structures (ASKIN; ZHOU, 1997), i.e. setting the cost ratio between the robot cost (R) and equipment cost (E). The chosen rations represent the practical case rate (approximately

$R/E = 2$), $R/E = 1$ (Equal: robots and equipment have comparable costs), $R/E = 30$ (High: robots are much more expensive than equipment), and $R/E = 1/30$ (Low: robots are much cheaper than equipment). Moreover, tools that are able to perform the same tasks in a reduced time are included in the equipment pool in order to evaluate computational complexity and a possible trade-off. Faster robots and tools combination are capable of executing the same tasks normal robots and tools do in 60% of the time, and cost twice as much. For instance, if a welding robot that performs a copy of task t in 10 time units and costs 10 monetary units ($) is considered, an additional welding robot that performs the same copy of task t in 6 time units and costs 20 monetary units ($) is also considered in the enlarged set.

Thus, the combination of both experiments (DT and R/E variations) resulted in a total of 64 instances that were summarised in Table 4 and Table 5, containing the total number of robots in system ($\#vTNR$), the number of opened and doubled stations ($\#bSo$ and $\#bSd$), the number of robots on track-motion devices ($\#bTM$), and the computational time in seconds.

Fixed parameters were defined based on practical characteristics of robotic welding assembly lines found in automotive industries. The desired CT is set to 1000 time units, the DT ranged from 0 to 70% of it, the number of maximum stations (NS) is gradually increased by the user depending on the DT proportion and varies from 13 to 19, the necessary time to use the track-motion to 10% of the CT. The number of tasks is set to 40 (13 geometry tasks, 4 stud tasks and 23 finishing tasks), the number of copies of tasks varies from 1 to 20 replicas. The duration time of each copy ranges from 21 to 77 time units, and this value is increased by 50% if the task is performed at a transporter station. The Supporting Information is available and contains detailed data concerning these instances.

Table 4 – Results for different relative dead times (DT) and cost ratios (R/E) with a reduced equipment pool. $\#vTNR$, $\#bSo$, $\#bSd$, and $\#bTM$ stand for total number of robots in system, the number of opened, the number of doubled stations, and the number of robots on track-motion devices, respectively.

DT (%)	Cost ratios (R/E): Practical | Equal | High | Low				
	$\#vTNR$	$\#bSo$	$\#bSd$	$\#bTM$	CPU Time (s)
0	22 | 22 | 22 | 22	11 | 11 | 11 | 11	1 | 1 | 1 | 1	0 | 0 | 0 | 0	29 | 20 | 23 | 28
10	25 | 25 | 25 | 26	13 | 13 | 13 | 13	0 | 0 | 0 | 0	0 | 0 | 0 | 0	52 | 39 | 76 | 67
20	28 | 26 | 26 | 28	13 | 13 | 13 | 13	0 | 2 | 3 | 1	0 | 4 | 3 | 2	21 | 43 | 22 | 22
30	30 | 29 | 29 | 32	13 | 13 | 13 | 15	4 | 3 | 5 | 3	0 | 4 | 2 | 6	39 | 71 | 127 | 82
40	32 | 33 | 32 | 33	13 | 15 | 13 | 15	4 | 3 | 5 | 3	3 | 6 | 2 | 6	13 | 31 | 41 | 22
50	36 | 36 | 36 | 37	15 | 15 | 15 | 15	9 | 4 | 6 | 3	0 | 5 | 1 | 6	55 | 183 | 729 | 387
60	39 | 39 | 39 | 43	15 | 15 | 15 | 15	7 | 7 | 8 | 3	2 | 2 | 1 | 6	36 | 30 | 33 | 30
70	46 | 48 | 46 | 51	19 | 19 | 19 | 19	9 | 5 | 9 | 3	1 | 4 | 1 | 6	18 | 18 | 27 | 22

Source: Michels *et al.* (2018b).

Table 5 – Results for different relative dead times (DT) and cost ratios (R/E) with an enlarged equipment pool. $\#vTNR$, $\#bSo$, $\#bSd$ and $\#bTM$ stand for total number of robots in system, the number of opened, the number of doubled stations and the number of robots on track-motion devices, respectively.

DT (%)	Cost ratios (R/E): Practical | Equal | High | Low				
	$\#vTNR$	$\#bSo$	$\#bSd$	$\#bTM$	CPU Time (s)
0	19 | 19 | 19 | 19	9 | 9 | 9 | 9	0 | 0 | 0 | 0	0 | 0 | 0 | 0	72 | 190 | 150 | 302
10	23 | 20 | 23 | 21	11 | 9 | 11 | 9	0 | 0 | 0 | 0	0 | 0 | 0 | 0	487 | 207 | 3600 | 450
20	26 | 26 | 26 | 28	11 | 13 | 13 | 13	1 | 2 | 3 | 1	0 | 4 | 3 | 2	3600 | 211 | 183 | 135
30	29 | 30 | 29 | 31	13 | 13 | 13 | 13	3 | 2 | 3 | 2	0 | 4 | 1 | 4	3600 | 2058 | 3600 | 129
40	32 | 33 | 32 | 33	13 | 15 | 13 | 15	4 | 3 | 5 | 3	3 | 6 | 2 | 6	3600 | 386 | 3600 | 220
50	36 | 35 | 36 | 35	15 | 15 | 15 | 15	9 | 3 | 7 | 3	0 | 6 | 1 | 6	3600 | 944 | 3600 | 1227
60	39 | 39 | 39 | 43	15 | 15 | 15 | 15	7 | 7 | 10 | 3	2 | 2 | 0 | 6	3600 | 3600 | 3600 | 518
70	44 | 45 | 41 | 43	17 | 17 | 15 | 15	8 | 5 | 8 | 3	2 | 4 | 1 | 6	3600 | 3600 | 3600 | 1899

Source: Michels *et al.* (2018b).

Out of the 32 cases from Table 4, all of them were solved to optimality, whilst only 18 out of 32 cases from Table 5 would result in optimal solutions within the time limit. On average, the cost is increased in 9.92% whenever there is an increase of 10% in the DT, except for the pace from 60% to 70%. In this last case, the cost is impacted with a 16.69% raise and it clearly attests that such relative unproductive times would result in impractical production systems.

The cost ratio experiment turned out as expected, validating the model for the practical cases: the line layout is completely changed depending on robot and equipment relative costs. For the cases in which the robots are much more expensive ($R/E = 30$), the line applies parallel stations more frequently in order to take advantage on productive time enlargement. On the other hand, the use of track-motion devices was more intensive for the opposite cases ($R/E = 1/30$), since the robots are much cheaper than the tools, the model decided to allocate the equipment mainly on platform stations, where there is no penalisation on task performing time.

Comparing Table 4 with Table 5, it is possible to state that computational times were highly affected by allowing more equipment options in the pool. However, the increase on the DT does not necessarily influence the solving time directly, and the instances in which the robot cost was much lower than the equipment cost (Low R/E) expressed that this class of parameters presents a reduced computational time to be solved. Moreover, the potential gains in saving design costs were analysed based on the cases that reached optimality, i.e. the 18 out of 32 instances presented in Table 5. On average, a potential economy of 1.11% was obtained and the larger difference was found to be a 7.57% (Low R/E and 0% of DT) cost reduction.

The optimal answer was proved for all the instances in Table 4. Still, for the enlarged equipment pool instances (Table 5), there was a gap between the best found answer (UB) and the best possible answer (LB) in 14 out of 32 cases. The gap for each instance is shown in Table 6.

For the instances that did not prove optimality in one hour of computational processing time, the average gap was 7.97%.

Table 6 – Gaps for different relative dead times (DT) and cost ratios (R/E) for the enlarged equipment pool instances.

DT (%)	Gap: $(UB - LB)/UB$			
	Practical R/E	Equal R/E	High R/E	Low R/E
0	0%	0%	0%	0%
10	0%	0%	5.06%	0%
20	1.87%	0%	0%	0%
30	1.37%	0%	9.90%	0%
40	1.66%	0%	5.00%	0%
50	11.95%	0%	20.54%	0%
60	2.93%	3.26%	8.71%	0%
70	10.86%	7.32%	21.18%	0%

Source: Michels *et al.* (2018b).

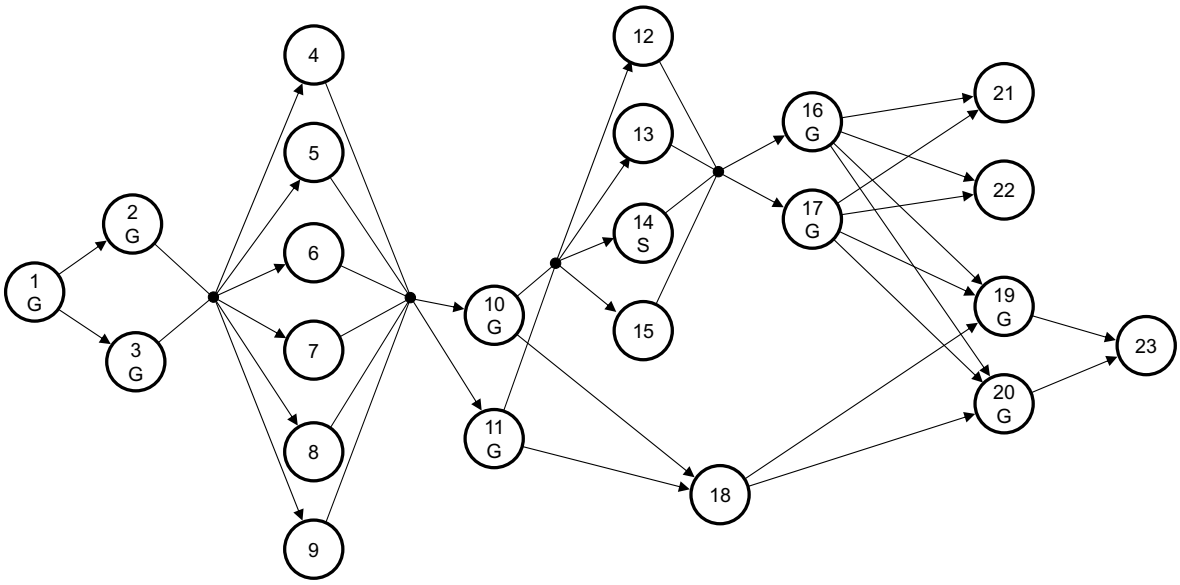
Lastly, the maximum number of stations (NS) for the herein reported tests were estimated based on previous knowledge of the problem and the proposed dataset, this NS was generally higher than necessary, i.e. a pessimistic estimation. Since all variable sets are built for all stations, an overestimation of this parameter may include unnecessary variables to the problem.

3.4.2 Practical Case Study

Currently, three different vehicle models are being produced in the studied line. The validated model explored in Section 3.4.1 has been employed to the data of each vehicle model in order to conduct the practical tests. Figure 12 shows the adapted real-world automotive industry precedence diagram presented by Sikora *et al.* (2017b), geometry and stud tasks are indicated in the diagram. Table 7 presents the task times for each vehicle model as long as they are performed in platform stations. These time durations are 50% longer if the task is performed in a transporter station. Geometry tasks are **bold-faced**, stud tasks are *italicised*, and the remaining ones are finishing tasks. Notice that less complex vehicles do not have all the assembly tasks that Model 3 does.

As for the other parameters, the operating line had been observed and movement time analysed to properly determine DT and TM (respectively 50% and 10% of the CT for the practical case), and the desired CT has also been informed by the company (1168 time units). The maximum number of stations is empirically estimated by measuring the possible

Figure 12 – Precedence diagram for all vehicle models. Geometry (G) and Stud (S) tasks are indicated in the diagram. The remaining ones are finishing tasks.



Source: Michels *et al.* (2018b), adapted from Sikora *et al.* (2017b).

Table 7 – Task times in platform stations for each model. Geometry tasks are boldfaced, stud tasks are in *italics*, and the remaining ones are finishing tasks.

Task	Model 1		Model 2		Model 3	
	Copies	Duration	Copies	Duration	Copies	Duration
1	**8**	**57**	**8**	**57**	**8**	**57**
2	**6**	**38**	**6**	**38**	**6**	**38**
3	**6**	**50**	**6**	**50**	**6**	**50**
4	6	47	4	49	10	42
5	10	29	10	29	20	27
6	14	58	10	57	6	55
7	18	40	18	40	22	43
8	4	47	4	47	8	43
9	4	63	4	63	2	64
10	**15**	**63**	**15**	**63**	**15**	**63**
11	**13**	**39**	**13**	**39**	**13**	**39**
12	7	42	11	38	11	38
13	34	35	46	34	40	37
14	*21*	*70*	*18*	*71*	*37*	*77*
15	11	28	7	26	13	21
16	**15**	**69**	**15**	**69**	**16**	**71**
17	**5**	**44**	**5**	**44**	**8**	**45**
18	0	-	0	-	12	37
19	**20**	**34**	**6**	**35**	**18**	**32**
20	**0**	**-**	**12**	**50**	**4**	**52**
21	12	35	12	35	14	33
22	12	55	12	55	12	51
23	0	-	6	56	11	52

Source: Michels *et al.* (2018b), adapted from Sikora *et al.* (2017b).

maximum length of the line and was set to $NS = 15$ for the practical case study. Industrial economic parameters have been collected, namely robot, equipment, track-motion, and platform

costs. These are not always the same for any project, they often depend on numeric studies and labour cost for installing the line. The price parameters are average normalised values ($) taken from the last recent projects and updates: platform cost ($PCost = 4.2$), track-motion cost ($TMCost = 10.3$), transporter robot cost holding no equipment other than the work-piece manipulation system ($RTCost_e = 19.8$), with a static welding tool placed in the sideways ($RTCost_e = 29.9$), with a static stud tool placed in the sideways ($RTCost_e = 25.8$), and platform robot cost holding a welding tool ($RPCost_e = 20.7$), or a stud tool ($RPCost_e = 18.4$). These prices and parameters can be found in the Supporting Information.

3.4.2.1 Line Design for Vehicle Models

Table 8 presents the results for the given parameters applied to each vehicle model. Naturally, the line cost is higher for Model 3, since it is the most complex vehicle model and has more assembly tasks than the other vehicle models.

Table 8 – Results for the three vehicle models produced by the company nowadays.

	Model 1	Model 2	Model 3
Cost ($)	557.5	561.5	628.6
$\#vTNR$	24	23	26
$\#bSo$	13	11	13
$\#bSd$	0	2	2
$\#bTM$	0	2	1
CPU Time (s)	40.1	31.2	32.7

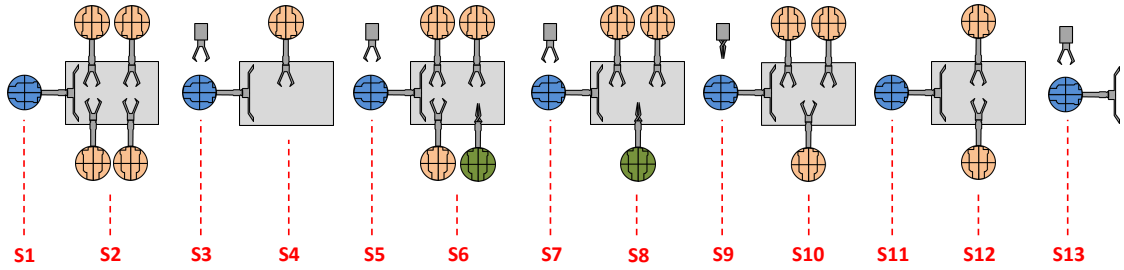
Source: Michels *et al.* (2018b).

The first and simplest vehicle model's layout configuration could be designed as an exclusively serial line in the optimal solution. Figure 13 shows the distribution of the robots and their tools' allocation through the stations.

Analogously to the representation of the first model, Figure 14 depicts the optimal layout configuration for Model 2. In this case, station paralleling has been employed in order to reduce costs for the design project and has also shorten the line's length. Moreover, track-motion devices are used to reach and move work-pieces in and out of parallel stations.

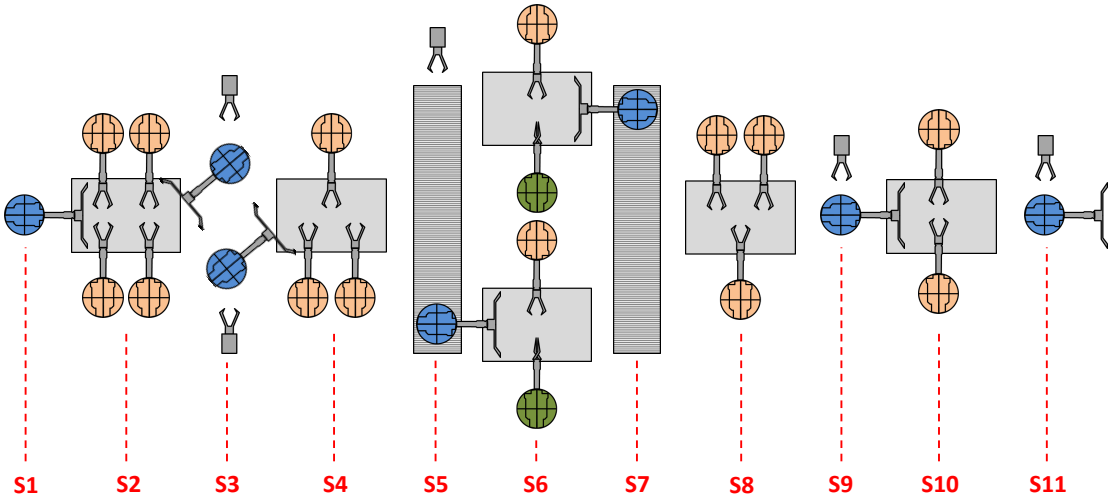
The optimal line design of the last and most complex vehicle model is shown in Figure 15. In this configuration, the production layout requires 11 serial stations, 2 parallel stations and a track-motion device. The Model 3's line employs more features than the first one and is longer than Model 2's line. This fact is expected due to the larger number of tasks and copies in the parameters.

Figure 13 – Optimal line design for Model 1. There are 13 serial stations (S1 to S13), no double stations or track-motion were employed on the configuration. There are 24 robots in total, composed of 17 platform robots (15 performing geometry and finishing welding tasks and 2 performing stud tasks) and 7 transporter robots (4 performing finishing welding tasks, 1 performing stud tasks (S9) and 2 for work-pieces handling, in the entrance and S11).



Source: Michels *et al.* (2018b).

Figure 14 – Optimal line design for Model 2. There are 11 stations (S1 to S11), 2 of them are doubled (S3 and S6) and 2 use a track-motion device (S5 and S7). There are 23 robots in total, composed of 16 platform robots (14 performing geometry and finishing welding tasks and 2 performing stud tasks) and 7 transporter robots (5 performing finishing welding tasks, none performing stud tasks and 2 for work-pieces handling, in the entrance and S7).

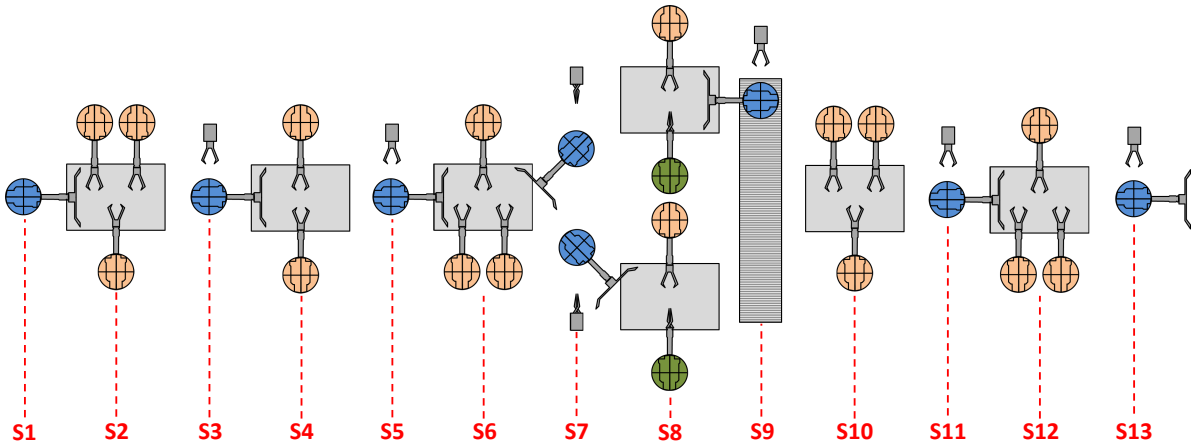


Source: Michels *et al.* (2018b).

3.4.2.2 Results Comparison

Figures 13, 14, and 15 represent robotic welding assembly lines for single products, as stated in the problem's hypotheses (Section 3.2). However, in the automotive industry, production systems are frequently built to process multiple models of vehicles, giving them the property of mixed-model assembly lines. Therefore, in order to analyse the applicability of any of these layouts, they must be feasible for all the vehicle models, otherwise, extra robots would have to be included in the faulty segments. Note that this approach can be seen as designing the line

Figure 15 – Optimal line design for Model 3. There are 13 stations (S1 to S13), 2 of them are doubled (S7 and S8) and 1 uses a track-motion device (S9). There are 26 robots in total, composed of 18 platform robots (16 performing geometry and finishing welding tasks and 2 performing stud tasks) and 8 transporter robots (5 performing finishing welding tasks, 2 performing stud tasks (S7) and 1 in the entrance for work-pieces handling).



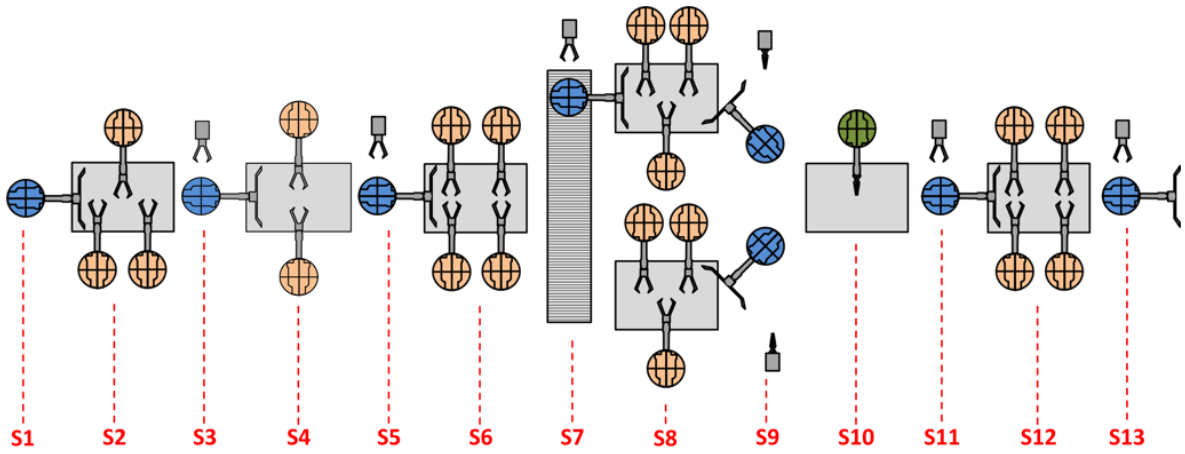
Source: Michels *et al.* (2018b).

for the worst case. In this situation, the layout proposed by the mathematical model for vehicle Model 3 is the only possible candidate to assume such position and is a natural candidate to be tested for the remaining vehicle models.

The adopted procedure was setting the variables in the optimisation model for the last vehicle model's design, apply it to the data of vehicle models 1 and 2 and verify its feasibility for each case. The obtained results indicate that the configuration presented in Figure 15 was able to support the production of the three vehicle models and, thus, allowing the cost comparison with the current as-built line, presented in Figure 16. Alternatively, (i) some robots could be disable depending on the vehicle model that is to be processed in order to avoiding idle times or (ii) the cycle time for the less complex models could be even reduced in specific situations. Nonetheless, it is important to state and remind that the costliest design for a single product is not necessarily fit to produce all the products in a mixed-model assembly line due to the task distribution possibilities and idle times caused by relative demands of the products. A more general approach for mixed-model lines is a future research goal (Section 3.5).

Table 9 presents a comparative between the model's solution for the optimal line design, the configuration proposed by the engineering team, and the strictly straight line for the Model 3. The optimal solution for Model 3 has been compared to the current as-built design and proven coherent, reinforcing the reliability of the mathematical formulation, previously stated by the computational results of Section 3.4.1. Similarly to the last procedure to test the Model 3's

Figure 16 – Current as-built line. There are 13 stations (S1 to S13), 2 of them are doubled (S8 and S9) and 1 uses a track-motion device (S7). There are 28 robots in total, composed of 20 platform robots (19 performing geometry and finishing welding tasks and 1 performing stud tasks) and 8 transporter robots (5 performing finishing welding tasks, 2 performing stud tasks (S9) and 1 in the entrance for work-pieces handling).



Source: Michels *et al.* (2018b).

layout to Models 1 and 2, the strictly serial line was simulated for the vehicle Model 3 by setting decision variables (bSd) to the desired values (always equal to zero). In other words, parallel stations had been forbidden in the model and it was applied to the vehicle Model 3's data. A similarity between the optimal solution and the as-built configuration might be noticed in Table 9. However, if Figure 15 is compared to Figure 16, one can realise that the optimal design did not just reduce the number of robots in the line, but also gave a different configuration from the current operating line as solution.

Table 9 – Comparative between the optimal line design, the configuration proposed by the engineering team (as-built), and the strictly straight line for Vehicle Model 3.

	Optimal	As-built	Serial
Cost ($)	628.6	667.7	692.4
$\#vTNR$	26	28	30
$\#bSo$	13	13	15
$\#bSd$	2	2	0
$\#bTM$	1	1	0

Source: Michels *et al.* (2018b).

The costs on Table 9 are normalised due to industrial reasons and do not appear to be so large in absolute values. The obtained relative values indicate a potential economy of approximately 5.9%, when comparing the as-built with the obtained optimal solution. Nonetheless, taking into consideration the purchase cost of industrial welding robots and the potential of

applying the model to all robotic lines found in an automotive industry, the cost reduction on the production layout can reach several hundred thousand dollars to be saved by the company.

Finally, this cost reduction comparison is only fair because both optimal and as-built layouts for Model 3 (Figure 15 and Figure 16, respectively) can produce all vehicle models through the same line and space, while respecting the demanded cycle time. Other layouts (Figure 13 and Figure 14) are completely valid (and optimal) for single vehicle model processing, but are not capable of producing all vehicle models under desired conditions of productivity rate (infeasible). Table 10 summarises which solution is optimal, feasible or infeasible for each model. Although there are optimal solutions for Models 1 or 2 for a lower cost, notice that such solutions are not feasible for the remaining vehicle models. Therefore, a global optimal solution could only be obtained by Layout 3 (Figure 15).

Table 10 – Feasibility verification between Models 1, 2, and 3 optimal layouts and as-built configuration.

	Model 1	Model 2	Model 3	Cost ($)
Layout 1 (Figure 13)	Optimal	Infeasible	Infeasible	557.5
Layout 2 (Figure 14)	Infeasible	Optimal	Infeasible	561.5
Layout 3 (Figure 15)	Feasible	Feasible	Optimal	628.6
As-built (Figure 16)	Feasible	Feasible	Feasible	667.7

Source: Michels *et al.* (2018b).

3.5 CONCLUSIONS

Robotic welding assembly lines are frequently found in the automotive industry and defining their production layout design is an important global and strategic decision. In this chapter, the Robotic Assembly Line Design (RALD) problem is defined and an MILP formulation is proposed for it, taking into account several practical considerations of an RALD scenario. The proposed model incorporates the linearisation of a cubic constraint. The developed model also allows to explicitly evaluate costs and benefits associated to parallel stations in an exact manner.

Computational case studies were performed in Section 3.4.1, combining large instances of real-world inspired cases adapted from Sikora *et al.* (2017b) and cost ratio principles proposed by Askin and Zhou (1997). The existence of multiple tool alternatives and with the trade-off between equipment cost and efficiency led to higher computational difficulties. However, 18 out of 32 of such cases were solved to optimality within the time limit (Table 5). The main conclusions drawn from this experiment are: (i) optimal answers tend to have more parallel stations as the dead time increases or the robots are costly compared to the equipment, and (ii)

the intense use of track-motion devices when equipment prices are much higher than the robot ones, due to its tendency to be more cost-effective.

Practical case studies based on the three vehicle models presented in Sikora *et al.* (2017b) reached optimal answers and led to a 5.9% cost reduction in the line design for the most complex model compared to the originally human-designed line (Section 3.4.2). This was only possible because the third vehicle model line layout was able to assemble both vehicle models 1 and 2, as indicated in Section 3.4.2.1. Furthermore, parallel stations evidenced its essential role when unproductive times are considered, though paralleling was not necessarily cost-effective in every condition (e.g. Figure 13).

Our study exposed how effective the formulation is when it comes to designing a robotic assembly line, including practical extensions. Therefore, for future research, the proposed model can be widened to incorporate task scheduling for each robot in the station. Moreover, the model might be adapted to represent literature variants, such as different product models characteristics in a mixed-model line and set-up times between them.

4 THE TYPE-1 MULTI-MANNED ASSEMBLY LINE BALANCING PROBLEM

This chapter contains a slightly modified version of the paper Michels *et al.* (2019), which is entitled "*A Benders' decomposition algorithm with combinatorial cuts for the multi-manned assembly line balancing problem*" and was published in the *European Journal of Operational Research*.

Section 4.1 introduces the type-1 multi-manned assembly line balancing problem (MALBP-1). Section 4.2 describes the studied problem, showing the main advantages of adopting multi-manned stations and its specific assumptions. Section 4.3.1 presents a new MILP model with symmetry breaks for the MALBP-1. Section 4.4 gives an overview of the proposed Benders decomposition algorithm (BDA) and how it is executed for the studied problem. Section 4.5 validates the results of the proposed BDA by optimally solving most MALBP-1 instances in the benchmark dataset. Final considerations of this chapter are presented in Section 4.6.

Abstract from Michels *et al.* (2019): *"Multi-manned assembly lines are commonly found in industries that manufacture large-size products (e.g. automotive industry), in which multiple workers are assigned to the same station in order to perform different operations simultaneously on the same product. Although the balancing problem of multi-manned assembly lines had been modelled before, the previously presented exact mathematical formulations are only able to solve few small-size instances, while larger cases are solved by heuristics or metaheuristics that do not guarantee optimality. This work presents a new Mixed-Integer Linear Programming model with strong symmetry break constraints and decomposes the original problem into a new Benders' Decomposition Algorithm to solve large instances optimally. The proposed model minimises the total number of workers along the line and the number of opened stations as weighted primary and secondary objectives, respectively. Besides, feasibility cuts and symmetry break constraints based on combinatorial Benders' cuts and model's parameters are applied as lazy constraints to reduce search-space by eliminating infeasible sets of allocations. Tests on a literature dataset have shown that the proposed mathematical model outperforms previously developed formulations in both solution quality and computational processing time for small-size instances. Moreover, the proposed Benders' Decomposition Algorithm yielded 117 optimal results out of a 131-instances dataset. Compared to previously presented methods, this translates to 19 and 25 new best solutions reached for medium and large-size instances, respectively, of which 19 and 23 are optimal solutions."*

4.1 INTRODUCTION

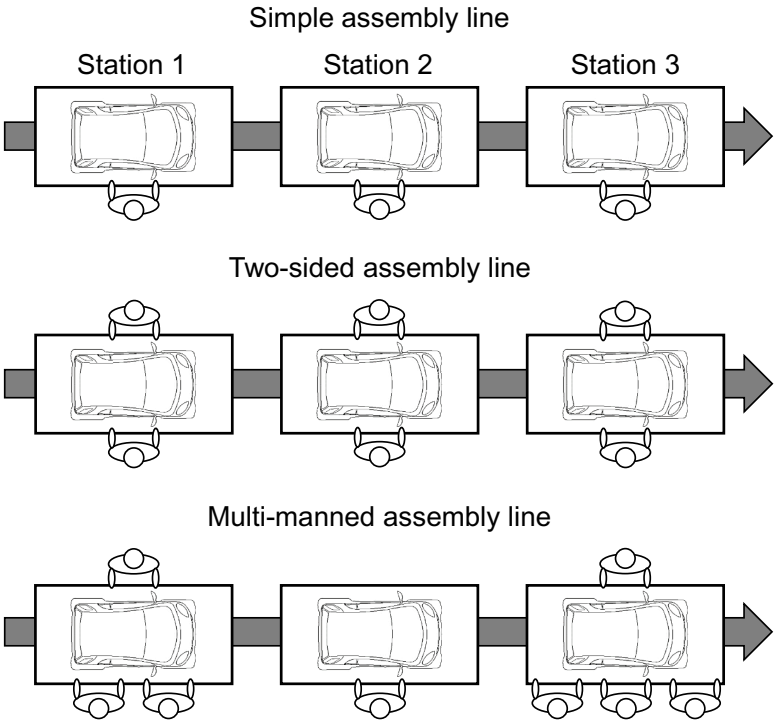
Production systems used in high-volume industries of standardised products are frequently based on flow-shop layouts, which are product-oriented designs. Assembly lines in a flow-shop configuration are generally dedicated to make homogeneous products, enabling their mass production. Along with this real-world usage, assembly lines have given rise to a combinatorial problem widely discussed in the literature (BATTAÏA; DOLGUI, 2013): the Assembly Line Balancing Problem (ALBP).

Considering several restrictive assumptions described by Baybars (1986), the problem of assigning a list of tasks subjected by a precedence graph to stations is called Simple Assembly Line Balancing Problem (SALBP). Allowing only one worker in each station is one of the restrictions. Moreover, these stations are organised in a straight, serial line that produces a unique model of a single product. The importance of ALBP is shown in the literature by the high number of published papers that still contribute to practical applications. In order to optimise (minimise) the number of stations (SALBP-1) or the cycle time (SALBP-2), several algorithmic solution methods were proposed: SALOME – an efficient bidirectional branch-and-bound procedure – was developed (SCHOLL; KLEIN, 1997) followed by a dynamic programming approach (BAUTISTA; PEREIRA, 2009), a branch, bound, and remember algorithm (SEWELL; JACOBSON, 2012), and an enhanced multi-Hoffmann heuristic (STERNATZ, 2014). These and other techniques were gathered in an overview and improved for SALBP-1 by Pape (2015).

However, assembly lines applied to automotive industry, for instance, commonly process large-size products, such as cars and buses. In these lines, the SALBP's hypothesis of allowing only one worker in each station often is not a practical limitation. As product size is rather large, it becomes admissible to assign more than one worker to each station and perform tasks simultaneously in different sectors of the same product, giving rise to natural extensions and more generalised versions of the SALBP: the Multi-manned Assembly Line Balancing Problem (MALBP) and the Two-sided Assembly Line Balancing Problem (TALBP), which are surveyed by Becker and Scholl (2006) along with a variety of practical extensions. Figure 17 depicts both above-mentioned lines, it shows simple, two-sided, and multi-manned assembly lines with three stations each. Nevertheless, two-sided and multi-manned assembly lines permit more than one worker in each station (i.e. stations 1 and 3), with workers performing tasks on the same product at the same time. The main difference between MALBP and TALBP is the flexibility on the

quantity of workers and their positioning. TALBP allows at most two operators, each of them at the station's right or left side, whereas in MALBPs the number of maximum workers depends on product's attributes, such as size, structure, and tasks' precedence relations. Another divergence is that TALBPs might have to deal with tasks that can be performed exclusively on the right or left side of the product.

Figure 17 – Configuration examples of a simple assembly line, a two-sided assembly line, and a multi-manned assembly line.



Source: Michels *et al.* (2019).

This work focuses on the MALBP variant; advantages of using the multi-manned configuration are associated to workforce and line length reduction, which are further exemplified in Section 4.2. Other simplification hypotheses from SALBP are kept, the most important ones to be mentioned are: (i) a straight, serial line is considered and (ii) the line produces a unique model of a single product.

In industrial environments, the use of multi-operated stations is intense. Consequently, numerous studies concerning MALBPs and TALBPs have recently been elaborated. To the best of the authors' knowledge, the Parallel Assignment Method (PAM) developed by Akagi *et al.* (1983) takes place as the first study in the literature to tackle the problem of achieving higher production rates in assembly lines with more than one worker in each station. Many years later, Dimitriadis (2006) proposed a heuristic method based on modifying a procedure created by Hoffmann (1963).

The heuristic has shown to be effective in enhancing space utilisation, with the objective of minimising the total number of workers and stations given a cycle time, which is still the most usual goal function employed in various works. Succeeding those papers, Becker and Scholl (2009) introduced the Assembly Line Balancing Problem with Variable Workplaces (VWALBP). In this problem, working areas are minimised given a cycle time, while work-pieces are divided into mounting positions and only a single worker is able to assemble them in each multi-manned station. A Mixed-Integer Linear Programming (MILP) model is generated including lower bounding techniques and a branch-and-bound algorithm named VWSolver (based on SALOME) is implemented to solve larger instances. Concomitantly, two-sided assembly lines were firstly explored by Bartholdi (1993), and its variants concerning mixed-model lines (ÖZCAN; TOKLU, 2009) and stochastic task times (ÖZCAN, 2010) were further developed.

Following those publications, works on MALBPs have been increasing yearly; Moon *et al.* (2009) included the feature of variably skilled workers into MALBPs, proposed a mathematical formulation, and solved large-size instances with a Genetic Algorithm (GA). Cevikcan *et al.* (2009) devised the application of multi-manned stations for mixed-model assembly lines with zoning constraints. Due to the complexity of the mathematical model, a five-phase heuristic was developed to solve it. However, none of them used exact algorithms, and their adopted methods would find near-optimal feasible solutions. The first mathematical model that minimises the total number of workers and stations simultaneously in a MALBP was proposed by Fattahi and Roshani (2011). They consider a single model line, in which the number of workers and stations are the primary and secondary objectives in the optimisation procedure, respectively. Their model could solve small-size instances in a reasonable amount of time, but failed in solving larger cases. For that reason, an Ant Colony Optimisation (ACO) algorithm has been developed to find feasible and near-optimal solutions for medium and large test problems. A novel efficient branch-and-bound algorithm called Jumper was developed by Kellegöz and Toklu (2012) to solve ALBPs with parallel multi-manned stations. Their algorithm outperforms the VWSolver in both quality of feasible solutions and computational processing times. Kazemi and Sedighi (2013) and Michels *et al.* (2018b) examine real-size cost-oriented problem instances for assembly lines with multi-operated stations: the first paper takes into consideration the objective of minimising total cost per production unit by presenting a heuristic method based on GA, whilst the latter one develops a MILP model to minimise design implementation costs (robots, station facilities, and equipment) of a robotic line that conceives the use of multiple

robots per station. Roshani *et al.* (2013) addressed the MALBP with a multi-objective function in their mathematical model, which maximises smoothness index and line efficiency, whereas minimising the line length. Moreover, an improved Simulated Annealing (SA) algorithm was proposed to solve the problem. Kellegöz and Toklu (2015) presented a constructive heuristic based on priority rules, a GA based improvement procedure, and conducted computational experiments on MALBP instances with the objective of minimising the total number of workers in the line. Yilmaz and Yilmaz (2015) aimed at the minimisation of number of workers, stations, and workload difference between workers with a mathematical formulation. Yilmaz and Yilmaz (2016a) also analysed the impacts of MALBPs with skilled workers and equipment needs, and for that a heuristic procedure was proposed for solving the problem. Roshani and Giglio (2017) approached the MALBP by trying to minimise the cycle time of a line as the primary objective, for a given number of stations. Besides the MILP model, two meta-heuristics based on SA algorithm were developed: the indirect and direct SA (ISA and DSA, respectively). The DSA performance in solving the problem showed to be better in terms of quality and computational time. In order to reduce the required workspace for shop operations, Chen (2017) developed a hybrid heuristic approach based on SA algorithms with specific practical extensions for the automotive industry, prioritising the minimisation of stations. Kellegöz (2017) has improved the mathematical formulation proposed by Fattahi and Roshani (2011) to minimise the total number of workers and stations in a MALBP. In addition, a Gantt-based heuristic is proposed within a SA algorithm to solve medium and large-size instances. This procedure outperforms the ACO algorithm presented by Fattahi and Roshani (2011) and finds better feasible solutions to most instances in the tested benchmark. Lastly, another SA algorithm is implemented by Roshani and Nezami (2017), this time to undertake the mixed-model MALBP with the minimisation of number of workers and stations as primary and secondary objectives, respectively.

Table 11 provides a summarised literature review and, by comparing the proposed method with previously published papers, it is possible to situate the proposed work's contribution into the literature. Although Becker and Scholl (2009) and Kellegöz and Toklu (2012) have developed exact methods to solve ALBPs with parallel workplaces, they had only considered the minimisation of working areas and the number of worker as a consequence, which is a different concept. The main concern from the literature appears to be problems with single model lines in which both total number of workers and stations are minimised. This is due to the reason that, as stated in several published articles (e.g. Fattahi and Roshani (2011), Kellegöz (2017), Roshani

and Nezami (2017)), minimising the number of workers might be more important reducing the number of stations. For that, mathematical models were developed along with heuristic (YILMAZ; YILMAZ, 2016a), genetic algorithm (MOON *et al.*, 2009), ant colony optimisation (FATTAHI; ROSHANI, 2011; YILMAZ; YILMAZ, 2016b), and simulated annealing (ROSHANI *et al.*, 2013; KELLEGÖZ, 2017) methods. However, none of these methods can guarantee optimality for medium and large-size instances. In order to fill such gap, a new mathematical formulation is developed with search-space reduction constraints and symmetry breaks to address the problem. Furthermore, a Benders' decomposition algorithm is proposed as an innovative exact method for the problem under study. By applying Benders' combinatorial cuts (BENDERS, 1962; CODATO; FISCHETTI, 2006) techniques as lazy constraints, larger benchmark instances can be solved to optimality. Differently from the classical Benders' decomposition, the proposed algorithm presents an integer slave problem intended for feasibility seeking. These works presented in Table 11, in particular Fattahi and Roshani (2011) and Kellegöz (2017), will serve as a benchmark for this chapter and the decomposition procedure herein proposed, which focus on minimising the total number of workers as the primary objective and the number of stations as the secondary one in a MALBP. In this way, a direct performance comparison of the objective function results is possible for each instance. Besides, these works (Fattahi and Roshani (2011) and Kellegöz (2017)) are the most recent ones concerning MALBPs with such minimisation objective and they also provide an extensive dataset to validate the proposed model and algorithm.

The remaining of the chapter is organised as follows. In Section 4.2, a deeper definition of MALBP is given in order to explain the problem. Section 4.3 presents the MILP model and the additional constraints to reduce the problem's search-space. Section 4.4 reviews the development and applications of Benders' decomposition and combinatorial cuts. It also describes in detail the proposed algorithm. Computational results retrieved from this study are presented and discussed in Section 4.5. Lastly, in Section 4.6, concluding remarks are summarised and further research directions are suggested.

4.2 PROBLEM STATEMENT

As mentioned, the assembly lines considered in this study are dedicated to mass production of a single model of a unique product. Their stations are positioned sequentially in a serial, straight line. Only one work-piece can be processed at a given time in each station. Contrary to unpaced and mixed-model lines, in which processing time oscillations, starvations, and blockages

Table 11 – Literature overview for the type-1 Multi-manned Assembly Line Balancing Problem (MALBP-1).

	Product diversity		Goal function				Solution method					
Author(s) (Year)	Single-model	Mixed-model	Production rate	Number of workers	Number of stations	Cost-oriented	Mathematical formulation	Heuristic	Genetic algorithm	Ant colony optimisation	Simulated annealing	Exact method
Akagi *et al.* (1983)	•		•					•				
Dimitriadis (2006)	•			•				•				
Becker and Scholl (2009)	•			•				•				•
Moon *et al.* (2009)	•			•	•		•		•			
Cevikcan *et al.* (2009)		•	•				•	•				
Fattahi and Roshani (2011)	•			•	•		•			•		
Kellegöz and Toklu (2012)	•			•								•
Kazemi and Sedighi (2013)		•				•	•		•			
Roshani *et al.* (2013)	•			•	•						•	
Kellegöz and Toklu (2015)	•			•			•	•				
Yilmaz and Yilmaz (2015)	•			•	•		•					
Yilmaz and Yilmaz (2016a)	•			•	•		•	•				
Roshani and Giglio (2017)	•		•				•				•	
Chen (2017)	•			•	•		•				•	
Kellegöz (2017)	•			•	•		•				•	
Roshani and Nezami (2017)		•		•	•		•				•	
Proposed work (2019)	•			•	•		•					•

Source: Michels *et al.* (2019).

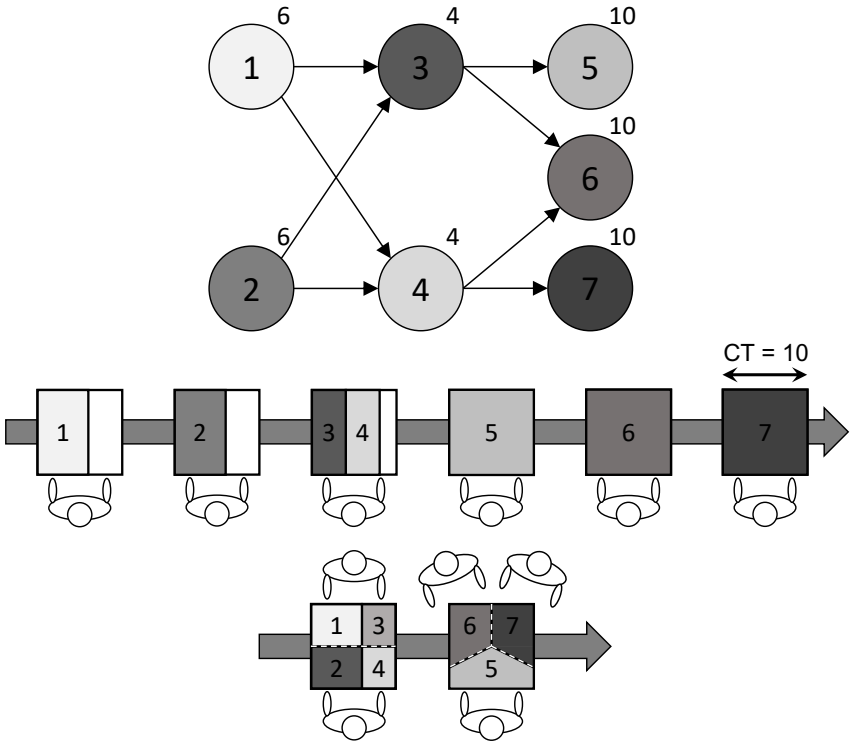
are relevant factors (LOPES *et al.*, 2018a), these pieces are moved forward between stations with a previously known and fixed cycle time (CT), while their transportation times are neglected. As the line produces a single product, its pace is exclusively determined by the most loaded station or the defined cycle time (BAYBARS, 1986).

In order to assemble any product, a set of tasks T must be performed. These tasks are indivisible and must respect precedence restrictions in their execution order. Each of them takes a deterministic duration time (D_t) to be completed, thus, the sum of these duration times assigned to the same worker must not exceed the defined cycle time. Nevertheless, due to parallel work within stations, it is also necessary for tasks to be scheduled in such a way that precedence relations are respected.

A sample instance is used to illustrate the difference of SALBP and MALBP. The precedence graph containing task indexes (number inside the circle) and durations (value on the top right corner of the circle), as well as optimal solutions for a SALBP and MALBP with a defined $CT = 10$ are presented in Figure 18. For the multi-manned line, each station is allowed to be occupied by more than one worker simultaneously working on the same work-piece. However,

the maximum number of operators (NW, with $NW = 3$ for the illustrative instance) admitted to perform different tasks concomitantly may vary due to the product size. The configuration of SALBP's optimal solution necessitates 6 workers assigned to 6 stations, totalling an idle time of 10 time units, or approximately 16.67% of the line's available working time, represented by the blank spaces in each station. On the other hand, by permitting more than one worker per station, the MALBP solution was able to not only reduce the line length from 6 to 2 stations, but also assign 5 workers instead of 6 to perform the task set. In this illustrative instance, it was possible to verify an advantage of multi-manned lines over simple ones by bringing the idle time down to zero. This efficiency improvement arises from allowing multiple workers to perform tasks simultaneously. Naturally, it depends on the instance's parameters.

Figure 18 – Precedence graph, SALBP-1, and MALBP-1 optimal solutions for the illustrative instance.



Source: Michels *et al.* (2019).

Nonetheless, this viable advantage comes along with the drawback of computational burden in solving the problem. Notice that the task indivisibility attribute is still valid and must be respected. Into the same station, each worker can only execute at most one task at a given time, and no cooperation is allowed between workers, i.e. no common task (YAZGAN *et al.*, 2011; SIKORA *et al.*, 2017a) can be performed by two or more workers together. Furthermore, tasks are not constrained by positioning and zoning restriction (BARTHOLDI, 1993; BECKER; SCHOLL,

2009), i.e. no interference occurs between workers during the assembly process (LOPES *et al.*, 2017). Lastly, there is no heterogeneity among workers (MOREIRA *et al.*, 2015), i.e. all workers have the same capacity and can perform a task with the same specific time required for its execution. Whilst different tasks can be performed by different workers synchronously, they still must satisfy all precedence relations imposed by the precedence graph. Therefore, a task scheduling problem arises for each station with conceivable waiting times for workers before or between the execution of tasks, making MALBPs more complex than SALBPs (FATTAHI; ROSHANI, 2011).

For the studied problem, it is assumed that the balancing decision is a long-term plan due to the high costs and line utilisation associated to it. That said, it is considered that the cost of a worker is much greater than the cost of opening an additional station, since each worker has some associated costs such as wages, equipment, labour regulations, among others. Hence, the primary objective of the mathematical model presented in Section 4.3 is to minimise the total number of workers, accompanied by the secondary objective of minimising the total number of stations, that is, the line length.

Ultimately, this work presumes that optimal SALBP solutions are feasible configurations for MALBPs. Naturally, the necessary number of workers (and stations) to achieve an optimal solution for the SALBP can be accepted as an upper bound for the MALBP, since SALBPs are more restrictive and assume the number of workers and stations to be the same in a given solution. Likewise, it is reasonable to adopt the upper bound for the total number of stations (NS) in a MALBP to be one unit lesser than its simpler counterpart optimal solution. As the MALBP's objective is to minimise both the number of workers and stations, with a higher weight in the former, the minimal marginal improvement taken from a SALBP solution is reducing the line length in one station by reallocating a worker in some of the remaining stations. For instance, the upper bound for the number of stations in a MALBP would be considered to be five ($NS = 5$) for the illustrative example presented in Figure 18, since that is the SALBP's optimal number of stations minus one. It is justified by the reasoning that, if the model is not even able to reduce one station in the previous solution by pointing out an infeasibility, then it is concluded that allowing more than one worker per station cannot contribute to efficiency improvements, and optimal solutions of both versions (SALBP and MALBP) are coincident in objective value. This fact is further analysed in Section 4.3.3.

4.3 MATHEMATICAL FORMULATION

This section contains a Mixed-Integer Linear Programming (MILP) model developed to represent the Multi-manned Assembly Line Balancing Problem (MALBP) considering the characteristics identified in Section 4.2. Section 4.3.1 presents the main model to represent the problem and Section 4.3.2 exhibits the implemented symmetry breaks that strengthens the problem's linear relaxation. Table 12 informs the applied terminology to describe parameters and sets used in the formulation. The variables are detailed in Table 13, they are created by the model depending on the sets.

Table 12 – Terminology: MALBP-1 names of parameters and sets, their meaning, and [dimensional units].

Parameter	Meaning
NT	Number of tasks
NS	Maximum number of stations
NW	Maximum number of workers per station
CT	Cycle time [time units]
D_t	Duration [time units] of task t
$WCost$	Worker cost [monetary units]
$SCost$	Station cost [monetary units]
$BigM$	A sufficiently large positive number
Set	**Meaning**
T	Set of tasks t; $T = \{1, 2,..., t,..., NT\}$
S	Set of stations s; $S = \{1, 2,..., s,..., NS\}$
W	Set of workers w; $W = \{1, 2,..., w,..., NW\}$
TS	Set of feasible Task-Station elements
TW	Set of feasible Task-Worker elements
WS	Set of feasible Worker-Station elements
TWS	Set of feasible Task-Worker-Station elements
P	Set of precedence relations between two tasks t_i and t_j: (t_i,t_j)

Source: Michels *et al.* (2019).

Table 13 – Terminology: MALBP-1 definition of model's variables.

Variable	Set	Domain	Meaning
$X_{t,s}$	$(t,s) \in TS$	$\{0,1\}$	Task-station assignment: set to 1 if task t is performed in station s
$Y_{w,s}$	$(w,s) \in WS$	$\{0,1\}$	Worker-station assignment: set to 1 if worker w is used in station s
$W_{t,w}$	$(t,w) \in TW$	$\{0,1\}$	Task-worker assignment: set to 1 if task t is performed by worker w
Z_s	$s \in S$	$\{0,1\}$	Station opened: set to 1 if station s needs to be used
F_{t_i,t_j}	$t_i, t_j \in T \mid t_i \neq t_j$	$\{0,1\}$	Follow variable: set to 1 if task t_i is followed by task t_j
ST_t	$t \in T$	$\mathbb{Z}_+$	Starting time: time in which task t starts to be performed
$A_{t,w,s}$	$(t,w,s) \in TWS$	$\{0,1\}$	Auxiliary variable: set to 1 for mimicking variables $Y_{w,s}$
$I_{w,s}$	$(w,s) \in WS$	$\mathbb{Z}_+$	Idle time: total time that worker w spends idle in station s

Source: Michels *et al.* (2019).

4.3.1 Main model

The objective function considered in Expression 25 for this problem is similar to the ones used in Fattahi and Roshani (2011) and Kellegöz (2017). The first component in the objective function corresponds to the total number of workers employed in line and their weighted cost ($WCost$). The remaining of the expression represents the total number of stations used in the line, along with its weighted cost ($SCost$). Remind that $WCost$ is a positive number much larger than $SCost$, therefore, the primary objective is to minimise the total number of workers.

$$\text{Minimise: } \underbrace{WCost \cdot \sum_{(w,s) \in WS} Y_{w,s}}_{workers\ cost} + \underbrace{SCost \cdot \sum_{s \in S} Z_s}_{stations\ cost} \tag{25}$$

In the model's constraints, Equations 26 are the occurrence constraint, forcing each task to be allocated to a station exactly once. Equations 27 are analogous to the previous one, it ensures that each task is exclusively performed by one worker. Equations 28 assign appropriate values to $Y_{w,s}$ variables: if a task t is allocated to station s, and this same task t is performed by worker w, then it is possible to induce which worker w from station s is employed for the activity. The precedence relations between tasks allocated in different stations are satisfied by Inequalities 29.

$$\sum_{s \in S} X_{t,s} = 1 \qquad \forall\, t \ \in \ T \tag{26}$$

$$\sum_{w \in W} W_{t,w} = 1 \qquad \forall\, t \ \in \ T \tag{27}$$

$$Y_{w,s} \geq X_{t,s} + W_{t,w} - 1 \qquad \forall\, (t,s) \ \in \ TS,\ (t,w) \ \in \ TW \tag{28}$$

$$\sum_{s \in S} s \cdot X_{t_i,s} \leq \sum_{s \in S} s \cdot X_{t_j,s} \qquad \forall\, (t_i,t_j) \ \in \ P \tag{29}$$

The task scheduling core of the problem is based on the concept of task following, i.e. when a task can only start after other is finished. How task following and task starting time variables behave in the formulation is hereafter presented. If a pair of tasks (t_i,t_j) is contained in the precedence set P, it is mandatory that task t_j follows task t_i (Equations 30). Logically, tasks with the same index cannot follow one another, and so is the set F_{t_i,t_j} accordingly defined. Inequalities 31 and 32 are logical ties to properly decide following variables depending on

task allocation and worker use. Between stations, if a task t_i is not allocated to the station that t_j is or in any station after that, then t_j follows t_i (Inequalities 31). Into the same station, if tasks t_i and t_j are performed by the same worker, then one of them must follow the other (Inequalities 32). Contrary to previously presented mathematical formulations that use a relative order time reasoning (FATTAHI; ROSHANI, 2011; KELLEGÖZ, 2017), this follow variable (F_{t_i,t_j}) concept allows the proposed model to be less dependent on Big-M based constraints. Nonetheless, the definition of task starting times still relies on few formulations containing Big-M strategies (HILIER; LIEBERMAN, 2015). In all cases, the $CT \cdot NS$ numerical value is a valid, sufficiently large, value for the parameter $BigM$ (Table 12). Inequalities 33 bind the starting time of a task t to a minimum value (lower bound) regarding station s in which it is allocated. From the other side, Inequalities 34 limit the maximum starting time (upper bound) that a task t can begin to be performed in station s. Complementary, the last task t must be finished up to the last opened station (Inequalities 35), and all tasks t_i that precede t_j must be completed before t_j starts to be performed (Inequalities 36).

$$F_{t_j,t_i} = 1 \qquad \forall\, (t_i,t_j) \;\in\; P \tag{30}$$

$$F_{t_j,t_i} \geq X_{t_j,s} - \sum_{sk \in S | sk \geq s} X_{t_i,sk} \qquad \forall\, t_i, t_j \;\in\; T,\; s \;\in\; S \tag{31}$$

$$F_{t_i,t_j} + F_{t_j,t_i} \geq X_{t_i,s} + X_{t_j,s} + W_{t_i,w} + W_{t_j,w} - 3 \qquad \forall\, (t_i,w,s), (t_j,w,s) \;\in\; TWS \mid t_i \neq t_j \tag{32}$$

$$ST_t \geq CT \cdot (s-1) - BigM \cdot (1 - X_{t,s}) \qquad \forall\, (t,s) \;\in\; TS \tag{33}$$

$$ST_t + D_t \leq CT \cdot s + BigM \cdot (1 - X_{t,s}) \qquad \forall\, (t,s) \;\in\; TS \tag{34}$$

$$ST_t + D_t \leq CT \cdot \sum_{s \in S} Z_s \qquad \forall\, t \;\in\; T \tag{35}$$

$$ST_{t_j} \geq ST_{t_i} + D_{t_i} - BigM \cdot (1 - F_{t_j,t_i}) \qquad \forall\, t_i, t_j \;\in\; T \mid t_i \neq t_j \tag{36}$$

4.3.2 Symmetry break constraints

The model 25–36 represents the problem. However, some ordering symmetry breaks were implemented into the model to strengthen the problem's linear relaxation and avoid wasting much time visiting symmetric solutions (WALSH, 2006). Inequalities 37 state that a station can only be opened if there is a task allocated to it. Inequalities 38 are similar, but for a worker assigned to that station. The combination of these inequalities assists the searching process for tighter bounds, as they prohibit the existence of unproductive or unoccupied stations. Inequalities 39 state that a station can only be opened if a previous one is already opened, they prevent the issue found by Yilmaz and Yilmaz (2016b) in a previous paper (FATTAHI; ROSHANI, 2011), in which an arbitrary opening order of stations was allowed, leading to inconsistent solutions. Analogously, Inequalities 40 break the symmetry between workers by stating that a worker can only be used if a previous one is already in use, avoiding equivalent solutions (in terms of objective function value) to be taken into consideration by the model and, therefore, shrinking the search-space. Notice that Inequalities 39 and 40 are only applied to the model from the second station/worker onwards, as it also was used by Kellegöz (2017).

$$Z_s \geq X_{t,s} \qquad \forall\, (t,s) \;\in\; TS \tag{37}$$

$$Z_s \geq Y_{w,s} \qquad \forall\, (w,s) \;\in\; WS \tag{38}$$

$$Z_s \leq Z_{s-1} \qquad \forall\, s \;\in\; S \mid s > 1 \tag{39}$$

$$Y_{w,s} \leq Y_{w-1,s} \qquad \forall\, (w,s) \;\in\; WS \mid w > 1 \tag{40}$$

Finally, idle time symmetry breaks are also added to the formulation. An auxiliary variable ($A_{t,w,s}$) that mimics the worker-station assignment ($Y_{w,s}$) is necessary for this part of the formulation. Inequalities 41 and 42 are the bounds to define appropriate values for $A_{t,w,s}$. This is necessary to calculate idle times associated to each worker along stations ($I_{w,s}$), which is done by Inequalities 43 and 44. Inequalities 45 conduct the symmetry break based on idle time information: the first worker's idle time must be lesser or equal to the second's and so on. This also enables the reduction of search-space, since equivalent solutions would be disregarded by the model due to rules concerning idle time differences between workers imposed by Inequalities 45.

$$A_{t,w,s} \geq X_{t,s} + W_{t,w} - 1 \qquad \forall\,(t,w,s)\ \in\ TWS \tag{41}$$

$$\sum_{(t,w,s)\in TWS} A_{t,w,s} \cdot D_t \leq CT \qquad \forall\,(w,s)\ \in\ WS \tag{42}$$

$$I_{w,s} \geq CT \cdot Y_{w,s} - \sum_{(t,w,s)\in TWS} A_{t,w,s} \cdot D_t \qquad \forall\,(w,s)\ \in\ WS \tag{43}$$

$$\sum_{(w,s)\in WS} I_{w,s} + \sum_{t\in T} D_t \leq CT \cdot \sum_{(w,s)\in WS} Y_{w,s} \tag{44}$$

$$I_{w,s} \geq I_{w-1,s} \qquad \forall\,(w,s)\ \in\ WS \mid w > 1 \tag{45}$$

The MILP formulation defined by 25–45 is henceforth referred to as PM (proposed model).

4.3.3 Upper bound value for NS

This section mathematically formalises the modelling decisions for upper bound values applied to NS. By hypothesis, the SALBP is a very restrictive problem, constrained by several simplification hypotheses (BAYBARS, 1986). One of them states that each station is operated by one worker. Thus, by minimising the number of stations, one is, in practice, minimising the number of workers as a consequence. When such hypothesis is relaxed, more than one worker can be allowed in each station, and the number of workers and stations are explicitly minimised separately. Furthermore, it is possible to attribute weights to workers and stations based on the importance (cost) of each resource, in which cumulative wages generally are much more costly than the physical parts of a station (FATTAHI; ROSHANI, 2011; KELLEGÖZ, 2017).

That stated, it is known that the number of workers and stations in both SALBP and MALBP cases are integers. For SALBP, x_S and y_S represent the number of workers and stations in a given configuration, respectively. Analogously, let x_M and y_M respectively represent the number of workers and stations in a given configuration of MALBPs. Naturally, x_S and y_S will always assume the same integer value ($x_S = y_S$) in any solution, since it is, by hypothesis, mandatory for a SALBP to have the same number of workers and station. Conversely, x_M is considered to be greater or equal to y_M ($x_M \geq y_M$), because each opened station must have at

least one worker performing operations in it. Moreover, the worker component weighted cost (w_1) is much larger than its station counterpart (w_2): $w_1 >> w_2$. Therefore, an objective function considering SALBP hypotheses can be expressed as $S(x_S,y_S) = w_1 \cdot x_S + w_2 \cdot y_S \mid x_S = y_S$ and a MALBP objective function as $M(x_M,y_M) = w_1 \cdot x_M + w_2 \cdot y_M \mid x_M \geq y_M$.

In terms of optimal solutions, the number of workers and stations are represented by x_S^*, y_S^*, x_M^*, and y_M^* for SALBP and MALBP cases, respectively. As MALBPs are less restrictive than SALBPs, their optimal solutions for the minimisation problem cannot be worse than their simpler counterpart in any given instance: Proposition 1.

Proposition 1. $M(x_M^*,y_M^*) \leq S(x_S^*,y_S^*)$*, which can be subdivided into two cases:*

1. $M(x_M^*,y_M^*) = S(x_S^*,y_S^*)$
2. $M(x_M^*,y_M^*) < S(x_S^*,y_S^*)$

Proof. In Proposition 1, case (i), the SALBP solution has, by hypothesis, the same number of workers and stations ($x_S^* = y_S^*$). Therefore, in order to have an optimal MALBP solution equivalent to a SALBP one, x_M^* must be equal to x_S^*, and y_M^* must be equal to y_S^*, that is: $M(x_M^*,y_M^*) = S(x_S^*,y_S^*) \Leftrightarrow x_M^* = x_S^* = y_M^* = y_S^*$, which is perfectly feasible. For case (ii), in which the optimal MALBP solution value of a given instance is exclusively lower than the optimal SALBP one, at least one of the following conditions must happen: (a) fewer workers ($x_M^* < x_S^*$), which would inevitably lead to fewer stations (a station without any worker/operation cannot be opened), or (b) same number of workers, but fewer stations ($x_M^* = x_S^* \wedge y_M^* < y_S^*$), which still meets the MALBP hypothesis of allowing more than one worker per station and is also reckoned possible by an illustrative example (Figure 18).

Given the assumption that the worker cost component receives a much higher weight in the objective function than the station one ($w_1 >> w_2$), it can be concluded by Proposition 1, case (ii), that the minimal marginal improvement in a MALBP solution over a SALBP one comes from the reduction of the line length in one station unit ($x_M^* \leq x_S^*$, $y_M^* < y_S^*$), and from case (i) that, in the worst case, the MALBP optimal solution is equivalent to the SALBP one, which produces the following corollary that is used in the proposed mathematical model.

Corollary. The upper bound for the number of stations (NS) in a MALBP model can be set to a value one unit lesser than the optimal solution to its SALBP counterpart (taking $w_1 >> w_2$ into account). If the model is found to be infeasible, it is concluded that the MALBP optimal solution is exactly the same as the configuration yielded by its SALBP version. □

4.4 BENDERS' DECOMPOSITION ALGORITHM

The Benders' decomposition algorithm (hereafter referred to as BDA) developed for the MALBP is presented in this section. The Benders' decomposition (BENDERS, 1962) is a method based on reformulating the original monolithic model into two hierarchical problems: the master problem (MP) and the slave (or sub) problems (SP). The partition aims at freeing the MP from several variables and restrictions, which are then solved as SPs. A Benders' decomposition works iteratively: a solution of the MP (with fixed values for the key variables) is then used in the SPs. The SP is generally decomposable in multiple problems, reducing the computational burden comparing to a monolithic problem. The results of such divided problems are used to inform the MP by using cutting planes. The MP with extra restrictions is solved and the procedure repeats with the next answer. In other words, the decomposition strips off difficult variables from the MP and then iteratively corrects misled solutions by solving the parts that are omitted in the MP. In the proposed implementation, the MP is related to the high-level decisions of task-station and worker-station assignments, whereas SP is associated to feasibility tests on the lower-level problem of task-worker assignment and task-scheduling in each station. Besides, the MP is enhanced by graph-based feasibility cuts described in Section 4.4.1. Extending the concept to use Combinatorial Benders' Cuts (CODATO; FISCHETTI, 2006), the MP is distilled from the original and complete combinatorial problem (monolithic model), which is equivalent to being significantly relaxed, since it is initially separated from the SP. Once all decisions variables to compute the objective value ($Y_{w,s}$ and Z_s) are contained in the MP, solving it might yield feasible integer solutions, which are sent to the SP to be validated (or not) by it. If feasibility is detected by the SP, the current solution is accepted as an incumbent one. Otherwise, the SP returns combinatorial inequalities to be added as lazy constraints into the MP. The BDA iterates this procedure until an optimal solution is found and proven.

The use of BDAs in the literature is reviewed and summarised by Rahmaniani *et al.* (2017). Several real-world problems were approached by using Benders' decomposition method. Nevertheless, when it comes to line balancing problems, only one work is listed in the survey (OSMAN; BAKI, 2014), which concerns transfer lines. Further research regarding assembly line balancing problems was scarcely developed. In Hazir and Dolgui (2013) and Hazir and Dolgui (2015), straight and U-type layouts are considered under uncertainty, formulating a robust optimisation model and algorithm. Lastly, Akpinar *et al.* (2017) takes into account set-up

times that are dependent on task sequencing in each station, interpreting task assignment and sequencing decisions as hierarchical problems. None of them involved assembly lines with multi-manned stations.

For the MALBP, the MP represents task-station assignments and worker-station allocation problems, whilst the SP takes care of the task-worker scheduling problem. Notice that, in this application, SP is not a continuous problem, thus a feasibility-seeking variant (BENDERS, 1962) must be used in order to solve the problem previously stated in Section 4.3 by the PM (Expressions 25–45). Therefore, the slave problem should be used as a feasibility check on the system, as it was stated by Côté *et al.* (2014) and Fakhri *et al.* (2017), who applied the method in the strip packing problem and the capacitated fixed charge multiple knapsack problem, respectively. In particular for the proposed BDA, the SPs are task-worker scheduling problems solved individually for each multi-manned station. As Benders decompositions might go through a slow convergence process (MAGNANTI; WONG, 1981), some algorithm enhancements are deemed necessary to accelerate such operation and so they are pointed out along with the MP and SP descriptions. Also for that reason, whenever an infeasibility is detected, such condition is modelled as a new restriction and added to the MP. These combinatorial Benders' cuts are further explained in Section 4.4.2. Moreover, feasibility cuts based on precedence graph analyses are implemented in the MP (Section 4.4.1), limiting the possibilities of task allocations.

4.4.1 Master problem

For the MP, Expressions 25, 26, 29, and 37–40 are maintained, and additional constraints are developed based on a set of incompatible task pairs (Inc), which are tasks that cannot be executed in the same station due to precedence relations and cycle time restrictions, independently on the number of workers assigned there (*Enhancement 1*). How this analysis is conducted to define such task pairs is hereafter presented. In order to do so, the precedence relations set P must be extended to a complete set P^* by considering all direct and indirect precedence relations. By constructing this complete set P^*, it is possible to create successors and predecessor sets for each task: Suc_t and Pre_t are sets that represent all direct and indirect successors and predecessors of a task t, respectively. Furthermore, two extra parameters are needed to represent the critical path duration (δ_{t_i,t_j}) and the sum of task durations (σ_{t_i,t_j}) between tasks t_i and t_j. When computed, these parameters are recursively evaluated in a topological order.

Given three tasks t_i, t_j, and t_k, such that $t_i \neq t_j \neq t_k$, Equations 46 recursively

attribute to parameter δ_{t_i,t_j} the critical path between two tasks by employing an algorithmic procedure, that is, they run through the precedence graph and establishes what is the longest sum of task durations that have to be performed between tasks t_i and t_j. This concept was adopted from project scheduling problems (KLEIN, 2000). Equations 47 are needed to further calculate capacity bounds, another logic inherited from the resource constrained project scheduling problems (KLEIN, 2000). The sum of task durations of all tasks that are successors of task t_i and predecessors of task t_j is assigned to the parameter σ_{t_i,t_j}. Taking the precedence graph from Figure 18 as an example, these parameters for the task pair (t_1,t_6) would be $\delta_{t_1,t_6} = 4$ and $\sigma_{t_1,t_6} = 8$.

$$\delta_{t_i,t_j} = \max\left[0;\ \delta_{t_i,t_k} + D_{t_k} \mid t_k \in Suc_{t_i} \cap Pre_{t_j}\right] \qquad \forall\ (t_i,t_j)\ \in\ P^* \tag{46}$$

$$\sigma_{t_i,t_j} = \sum_{t_k \in Suc_{t_i} \cap Pre_{t_j}} D_{t_k} \qquad \forall\ (t_i,t_j)\ \in\ P^* \tag{47}$$

In order to define which task pairs (t_i, t_j) are incompatible, only task pairs $(t_i, t_j) \in P^*$ are considered. If either Inequalities 48 or Inequalities 49 are verified, the incompatibility condition is satisfied, and the task pair (t_i, t_j) are added to the incompatibility set Inc. Once in hand of Inc, Inequalities 50 are added to the MP, restricting specific task pairs to be allocated to the same station. Whenever Inequalities 49 are not satisfied, it is possible to generate weaker – but still valid – restrictions. Inequalities 51 state the minimum number of workers that a station requires in order to perform both tasks of a task pair (t_i, t_j), being ε a very small positive number to avoid dividing by zero. Finally, the available time to perform tasks in each station is given by the number of workers assigned there (Inequalities 52).

$$D_{t_i} + D_{t_j} + \delta_{t_i,t_j} > CT \tag{48}$$

$$D_{t_i} + D_{t_j} + \left\lceil \frac{\sigma_{t_i,t_j}}{NW} \right\rceil > CT \tag{49}$$

$$X_{t_i,s} + X_{t_j,s} \leq 1 \qquad \forall\ s\ \in\ S,\ (t_i,t_j)\ \in\ Inc \tag{50}$$

$$\sum_{w \in W} Y_{w,s} \geq \left\lceil \frac{\sigma_{t_i,t_j}}{CT - D_{t_i} - D_{t_j} + \varepsilon} \right\rceil - NW \cdot (2 - X_{t_i,s} - X_{t_j,s}) \qquad \forall\ s\ \in\ S,\ (t_i,t_j)\ \in\ P^* \tag{51}$$

$$\sum_{t \in T} X_{t,s} \cdot D_t \leq CT \cdot \sum_{w \in W} Y_{w,s} \qquad \forall\, s \ \in \ S \tag{52}$$

This reformulated part of BDA focus on finding an optimal solution for the problem. Any feasible solution $(\tilde{X}, \tilde{Y}) = \{(\tilde{X}_{1,1}, ..., \tilde{X}_{t,s}), (\tilde{Y}_{1,1}, ..., \tilde{Y}_{w,s})\}$ found by the MP is passed to SP for scheduling feasibility check in each station s. That way, the monolithic model is decomposed in an MP that decides allocation variables ($X_{t,s}$) and the number of total workers and stations ($Y_{w,s}$ and Z_s) used along the line, while the SP seeks for feasible task-worker assignments ($W_{t,w}$) by considering task starting times and ordering (ST_t and F_{t_i,t_j}) for each station.

4.4.2 Slave problem

The SP keeps Expressions 27, 30, and 36 as in Section 4.3, but modifies Inequalities 32, 43, and 45 for simpler ones. For that, they are applied to each station s separately by using task and worker sub-sets T_s and W_s, which are dependent on the solution $(\tilde{X}, \tilde{Y})$ sent from MP, as expressed by Equations 53 and 54. Thinking of each station as a separate resource-constrained scheduling sub-problem, Inequalities 55, 56, and 57 respectively substitute the previous monolithic ones without any loss of functionality. Slave problems in which only one worker is employed in the station are automatically deemed feasible, since there is no scheduling problem to begin with in such cases.

$$T_s = \{t \ \in \ T \mid \tilde{X}_{t,s} = 1\} \qquad \forall\, s \ \in \ S \tag{53}$$

$$W_s = \{w \ \in \ W \mid \tilde{Y}_{w,s} = 1\} \qquad \forall\, s \ \in \ S \tag{54}$$

$$F_{t_i,t_j} + F_{t_j,t_i} \geq W_{t_i,w} + W_{t_j,w} - 1 \qquad \forall\, t_i, t_j \ \in \ T_s \tag{55}$$

$$I_w = CT - \sum_{t \in T_s} D_t \cdot W_{t,w} \qquad \forall\, w \ \in \ W_s \tag{56}$$

$$I_w \geq I_{w-1} \qquad \forall\, w \ \in \ W_s \mid w > 1 \tag{57}$$

In order to prevent redundant feasibility seeking tests, a hash-table (MAURER; LEWIS, 1975) is employed to store SP instance information that had led to feasible solutions (*Enhancement 2*). Whenever a task allocation set for a given station is tested and found to be feasible, that set of tasks and number of workers used to perform them is included into a tested problem hash-table, so the SP model does not need to solve repeated scheduling problems, since the algorithm can quickly consult this hash-table beforehand. If the MP's solution is evaluated as feasible for all stations by the SP, then such solution is considered incumbent by the proposed BDA and the algorithm returns to the MP with a new UB. Otherwise, the submitted MP's solution might be detected as infeasible. If that is the case, Benders' combinatorial cuts are applied to the MP as lazy constraints, that is, new restrictions are added to the initial optimisation problem in order to cut off infeasible task allocation and worker assignment possibilities (*Enhancement 3*).

The type of cut depends on the number of workers that the SP instance had employed and was proven infeasible. Stronger cuts can be added when the trial solution employs the maximum allowed workers for a specific station. Inequalities 58 state that, if a given tested task allocation set cannot be performed in the same station s, then at least one of them must be performed elsewhere. Alternatively, the allocation may be infeasible, but the maximum number of workers is not being used for that combination of tasks. In such cases, the cut described by Inequalities 59 are applied to the MP; it states that a tested task allocation set cannot be entirely performed in the same station unless an additional worker (represented by the $|W_s| + 1$ index) is assigned there.

$$\sum_{t \in T_s} X_{t,s} \leq \sum_{t \in T} \tilde{X}_{t,s} - 1 \qquad \forall\ s\ \in\ S \tag{58}$$

$$\sum_{t \in T_s} X_{t,s} \leq \sum_{t \in T} \tilde{X}_{t,s} - 1 + Y_{|W_s|+1,s} \qquad \forall\ s\ \in\ S \tag{59}$$

After solving all SPs, the BDA returns to MP and keeps searching for better solutions with a revised UB or newly added lazy constraints. This process is repeated iteratively until an optimal solution is found and proven or the computational processing time limit is reached.

4.4.3 BDA pseudo code

A summarised pseudo-code of the proposed BDA is presented in Algorithm 1. This is the implementation used to obtain the results reported in the computational study performed in Section 4.5. Initially, MALBP's parameters and computational processing time limit are input in

order to build the optimisation problem. After that, the algorithm starts solving the MP (line 5). Tight upper bounds for the number of workers along the line ($ObjW$) and the number of opened stations ($ObjS$) based on SALBP results are taken into consideration to shrink the search-space (*Enhancement 4*). Any time a new solution is found by MP, it is sent to feasibility check at the SP (line 6). Task-worker scheduling is conducted for each station (lines 8 and 9). When the combination of tasks and number of workers is feasible for a given station, such combination is added to a list coded as a hash-table (lines 10 and 11). Otherwise, if an infeasibility is detected, either Cut 58 or Cut 59 is added to the MP as a lazy constraint depending on the cardinality of W_s (lines 14 to 18). If all SP stations are proven to be feasible, this tested solution is considered to be the current incumbent solution (lines 21 and 22). Finally, the algorithm stops processing if optimality is proven or if time limit is reached (lines 23 and 24).

Algorithm 1 – BDA's pseudo-code for the MALBP-1.

1: **function** BDA($NT, NS, NW, CT, D_t, P, Time.Limit$)
2: $Status \leftarrow 0$, $Incumbent \leftarrow \{max.value, max.value, 0, -\}$, $Hash \leftarrow \{\}$, $Counter \leftarrow 0$
3: **while** $Status = 0$ **do**
4: Timer.Start, compute CPU time
5: Solve MP, compute $ObjW, ObjS, LB, TWS_{allocation}$
6: **if** $(\tilde{X},\tilde{Y})_{new}$ **then**
7: $Counter \leftarrow 0$
8: **for each** $s \in \{1,...,ObjS\}$ **do**
9: Solve SP $(\tilde{X},\tilde{Y})_s$, compute feasibility
10: **if** $feasible$ **then**
11: $Hash \leftarrow Hash \bigcup \{(\tilde{X},\tilde{Y})_s\}$
12: $Counter \leftarrow Counter + +$
13: **else**
14: **if** $|W_s| = NW$ **then**
15: Add Cut 58 to MP
16: **else**
17: Add Cut 59 to MP (in case $|W_s| < NW$)
18: **end if**
19: **end if**
20: **end for each**
21: **if** $Counter = ObjS$ **then**
22: $Incumbent(ObjW, ObjS, LB, TWS_{allocation}) \leftarrow (\tilde{X},\tilde{Y})_{new}$
23: **if** $(ObjW + ObjS = LB) \vee (CPU \geq Time.Limit)$ **then**
24: $Status \leftarrow 1$
25: **end if**
26: **end if**
27: **return** BDA($ObjW, ObjS, LB, CPU, TWS_{allocation}$)
28: **end while**
29: **end function**

Source: Michels *et al.* (2019).

4.5 COMPUTATIONAL STUDY

This section presents a computational study that was carried out in the same benchmark dataset used by Fattahi and Roshani (2011) and Kellegöz (2017) combined. Thus, both datasets were used, totalling 131 instances. All tested instances are contained in a well-known literature benchmark. They are available for download at <www.assembly-line-balancing.de> with information about task durations, precedence graphs, and cycle time values. Some of these instances' features can be observed in Table 14: the number of tasks (NT) is the chosen parameter to divide instances into three categories related to size (small, medium, and large), and instances are solved with different cycle time (CT) values and maximum number of workers (NW) allowed in each station. Upper bounds for the number of stations (NS) for each instance has been obtained in a pre-processing step: (i) the SALBP version of each problem is solved using SALOME (SCHOLL; KLEIN, 1997), which takes less than a second; (ii) NS is set to one unit lesser than such value for the MALBP. In order to ease results' visualisation and respect parameters' order of magnitude, $WCost$ and $SCost$ were set to 100 and 1, respectively.

Table 14 – Summary of MALBP-1 dataset instances.

Size (Total of instances)	Problem	NT	CT	NW
Small (50)	Mitchell	21	14; 15; 21; 26; 35; 39	2
	Heskiaoff	28	138; 205; 216; 256; 324; 342	2; 4
	Sawyer	30	25; 27; 30; 36; 41; 54; 75	2; 4
	Kilbridge	45	57; 79; 92; 110; 138 184	2; 4; 6
Medium (45)	Tonge	70	176; 364; 410; 468; 527	2; 4; 6
	Arcus1	83	5048; 5853; 6842; 7571; 8412; 8998; 10816	2; 4; 6
	Mukherje	94	176; 248; 351	2; 4; 6
Large (36)	Arcus2	111	5755; 8847; 10027; 10743; 11378; 17067	2; 4; 6
	Barthol2	148	84; 106; 170	2; 4; 6
	Barthold	148	403; 513; 805	2; 4; 6

Source: Michels *et al.* (2019).

The computational study is divided in two parts. Firstly, small-size instances are solved in Section 4.5.1 for the monolithic model presented in Section 4.3 (PM) and results are compared to those obtained by Kellegöz (2017); this last model is henceforth referred to as KM (Kellegöz's Model). In addition, the BDA exhibited in Section 4.4 is also applied to the same fraction of the benchmark dataset and its performance is compared to the Ant-Colony Algorithm (ACO) and the Gantt Simulated Annealing (GSA) heuristic developed by Fattahi and Roshani (2011) and Kellegöz (2017) in terms of solution quality reported by them.

Afterwards, as both BDA, ACO, and GSA demonstrated dominance over monolithic models in terms of solution quality and computational processing time, mathematical formula-

tions (PM and KM) were discarded in the remainder testing process. Therefore, for the second part of this computational study, only BDA, ACO, and GSA were considered to be applied to the remaining dataset (medium and large-size instances). Such results and comparisons are presented in Section 4.5.2.

In both Sections 4.5.1 and 4.5.2, each instance result is reported in a line of Tables 15 to 17, addressed by the problem, CT, and NW values. St indicates solution status, reporting optimal (*) or integer (I) solutions. The objective function value is represented by the Obj column. For the BDA, this Obj column is branched into upper bound (UB), lower bound (LB), and Gap values. Total computational processing times (CPU) and computational processing times for the slave problems (SCPU) are informed in seconds. As CPU of some instances were not reported in Fattahi and Roshani (2011), they are left in unfilled (–). Lastly, the number of added cuts (Cut1 and Cut2 for Inequalities 58 and 59, respectively) and the number of times that the algorithm accessed the hash-table (HT) are reported.

To all instances, Gurobi 8.1 (Gurobi Optimization, 2019) was selected as universal solver due to implementation readiness, focusing on optimality for the MP and feasibility for the SP. A 64 bit Intel™ i7-3770 CPU (3.4 GHz) with 16 GB of RAM was employed using four threads. The BDA was coded in Microsoft Visual Basic 2015 programming language.

4.5.1 Small-size instances

Both PM and BDA were applied to the small-size instances presented in Table 14 with a time limit set to 3600 seconds. Table 15 summarises the comparison between monolithic models and algorithms for the small-size dataset. The results obtained by PM are compared to those reported by KM whenever such instance has also been solved by Kellegöz (2017), whilst BDA results are displayed alongside with the best result found by either ACO or GSA in their respective papers. This subset contains 50 instances, in which PM clearly outperforms KM. The PM obtained 46 optimal solutions, whereas in the 26 instances tested by Kellegöz (2017), KM reached optimality in only 12 instances. In other words, PM has improved and proven the optimality of 6 solutions (boldfaced in Table 15) and proven the optimality of 7 previously known integer solutions (italicised in Table 15) obtained by a mathematical model. As reported in Section 4.3, this might be due to the fact that modelling decisions were different between PM and KM: the formulation is less dependent on Big-M constraints, a follow variable concept was employed instead of a relative order time one, and symmetry break constraints were

implemented.

BDA, ACO, and GSA results are reported in the remaining of the comparison by Table 15. Equivalent solutions and computational processing times are verified for both methods, however, the BDA presented in Section 4.4 is able to concede optimality proofs. Therefore, the BDA not only has reached results as good as the ACO algorithm and the GSA heuristic, but it also has guaranteed solutions to be optimal for 49 out of 50 small-size instances in a very reduced CPU time.

Table 15 – Results comparison for small-size instances between monolithic models (PM and KM), BDA, ACO algorithm and GSA heuristic.

Problem	*CT*	*NW*	PM			KM			BDA								ACO/GSA	
			St	Obj	CPU	St	Obj	CPU[1]	St	UB	LB	Gap	CPU	SCPU	Cut1	Cut2	HT	Obj	CPU[1]
Mitchell	14	2	*	807	0.09	*	807	3.40	*	807	807	0%	0.01	0.00	0	0	0	807	0.88
	15	2	*	807	0.08	–	–	–	*	807	807	0%	0.01	0.00	0	0	0	807	–
	21	2	*	505	0.17	–	–	–	*	505	505	0%	0.01	0.00	0	0	0	505	–
	26	2	*	504	0.06	*	504	0.44	*	504	504	0%	0.01	0.00	0	0	0	504	0.93
	35	2	*	303	0.05	–	–	–	*	303	303	0%	0.01	0.00	0	0	0	303	–
	39	2	*	302	0.04	*	302	0.44	*	302	302	0%	0.01	0.00	0	0	0	302	0.94
Heskiaoff	138	2	*	*805*	42.26	I	805	1h	*	805	805	0%	1.40	1.37	575	0	86	805	0.95
		4	*	804	0.80	*	804	19.98	*	804	804	0%	0.01	0.01	0	0	0	804	1.26
	205	2	*	503	0.36	–	–	–	*	503	503	0%	0.73	0.71	87	0	2	503	–
		4	*	503	2.23	–	–	–	*	503	503	0%	0.73	0.72	0	45	0	503	–
	216	2	*	503	0.38	–	–	–	*	503	503	0%	0.19	0.18	72	0	7	503	–
		4	*	503	1.91	–	–	–	*	503	503	0%	0.03	0.02	0	0	0	503	–
	256	2	*	**403**	11.10	I	503	1h	*	403	403	0%	1.59	1.46	1326	0	0	403	1.04
		4	*	**403**	10.12	I	502	1h	*	403	403	0%	2.78	2.50	0	1401	0	403	1.21
	324	2	*	402	0.07	–	–	–	*	402	402	0%	0.02	0.02	0	0	0	402	–
		4	*	402	0.07	–	–	–	*	402	402	0%	0.02	0.01	0	0	0	402	–
	342	2	*	302	0.28	*	302	53.43	*	302	302	0%	0.25	0.24	0	0	0	302	1.07
		4	*	302	0.49	*	302	17.04	*	302	302	0%	0.32	0.31	0	8	0	302	1.21
Sawyer	25	2	*	*1408*	2.42	I	1408	1h	*	1408	1408	0%	0.03	0.01	8	0	0	1408	1.46
		4	*	*1408*	3.94	I	1408	1h	*	1408	1408	0%	0.20	0.17	0	272	49	1408	1.79
	27	2	*	*1308*	14.34	I	1308	1h	*	1308	1308	0%	0.04	0.02	32	0	3	1308	1.19
		4	*	*1308*	9.06	I	1308	1h	*	1308	1308	0%	0.02	0.01	0	0	0	1308	1.52
	30	2	*	1206	25.38	–	–	–	*	1206	1206	0%	0.06	0.02	14	0	0	1207	–
		4	*	1206	73.25	–	–	–	*	1206	1206	0%	0.10	0.01	0	0	0	1207	–
	36	2	*	1006	8.26	–	–	–	*	1006	1006	0%	0.01	0.01	0	0	0	1006	–
		4	*	1006	6.99	–	–	–	*	1006	1006	0%	0.01	0.01	0	0	0	1006	–
	41	2	*	804	1.60	–	–	–	*	804	804	0%	0.07	0.01	0	0	1	806	–
		4	*	804	2.49	–	–	–	*	804	804	0%	0.05	0.03	0	0	0	806	–
	54	2	*	*704*	54.34	I	704	1h	*	704	704	0%	0.03	0.02	0	0	0	704	1.13
		4	*	*704*	5.88	I	704	1h	*	704	704	0%	0.01	0.01	0	0	0	704	1.44
	75	2	*	503	0.17	*	503	9.42	*	503	503	0%	0.01	0.01	0	0	0	503	1.28
		4	*	503	5.88	*	503	5.99	*	503	503	0%	0.01	0.01	0	0	0	503	1.70
Kilbridge	57	2	*	*1006*	3184.13	I	1006	1h	*	1006	1006	0%	3.43	3.15	390	0	20	1006	1.53
		4	*	**1005**	6.66	I	1105	1h	*	1005	1005	0%	0.53	0.48	5	50	9	1005	2.21
		6	*	1005	5.07	*	1005	2843.85	*	1005	1005	0%	3.02	2.96	0	45	3	1005	2.22
	79	2	*	704	27.90	–	–	–	*	704	704	0%	1.03	0.70	165	0	19	705	–
		4	I	803	1h	–	–	–	*	703	703	0%	0.33	0.28	0	0	0	705	–
		6	I	803	1h	–	–	–	*	703	703	0%	2.36	2.20	0	50	0	705	–
	92	2	I	604	1h	–	–	–	*	604	604	0%	93.82	91.75	4536	0	866	604	–
		4	*	603	8.82	–	–	–	*	603	603	0%	0.61	0.57	0	0	0	604	–
		6	*	603	48.31	–	–	–	*	603	603	0%	0.54	0.46	0	4	0	604	–
	110	2	*	603	0.89	*	603	663.27	*	603	603	0%	0.75	0.72	3	0	0	603	1.95
		4	*	603	1.66	*	603	323.27	*	603	603	0%	0.03	0.02	0	0	0	603	2.60
		6	*	603	2.07	*	603	63.88	*	603	603	0%	0.05	0.04	0	0	0	603	2.73
	138	2	I	403	1h	–	–	–	I	403	402	0.25%	1h	3551.78	987	0	1	403	–
		4	*	402	13.46	–	–	–	*	402	402	0%	10.41	10.39	0	3	0	403	–
		6	*	402	42.44	–	–	–	*	402	402	0%	21.50	21.45	0	12	0	403	–
	184	2	*	**302**	4.13	I	402	1h	*	302	302	0%	0.53	0.51	4	0	0	302	2.15
		4	*	**302**	9.29	I	402	1h	*	302	302	0%	0.66	0.64	0	4	0	302	4.25
		6	*	**302**	14.71	I	402	1h	*	302	302	0%	0.47	0.44	0	24	0	302	3.84

[1]As reported in Kellegöz (2017).

Source: Michels *et al.* (2019).

4.5.2 Medium and large-size instances

As BDA has been validated as a reliable and efficient method in Section 4.5.1 by quickly obtaining optimal solutions in all but one instance, the superiority of the proposed algorithm over the monolithic model was evidenced. Hence, this section focuses on computationally solving medium and large-size instance only with specialised methods (i.e. BDA, ACO, and GSA heuristic) and comparing their results in regard to solution quality that were previously reported in Fattahi and Roshani (2011) and Kellegöz (2017).

Table 16 reports the results for 45 medium-sized instances from Table 14. In terms of solution quality, the proposed BDA has outperformed ACO algorithm and GSA heuristic in 19 instances (boldfaced values in Table 16), while tying in the remaining 26. Nonetheless, it is important to notice that none of these instances had been directly solved to optimality previously, since their results would rather be compared to calculated theoretical LBs. Out of the 45 medium-sized instances, the proposed BDA has proven the optimality of 40 solutions, with a small integer gap for the remaining 5 cases.

The last 36 instances to be tested from Table 14 are contained in the large-size subset. Table 17 presents the comparison between BDA, ACO algorithm, and GSA heuristic when both are applied to such instances. The boldfaced values represent the 23 out of 36 results in which BDA has outperformed ACO and GSA in terms of solution quality and also proven optimality for the instance. Moreover, 2 previous best-known integer solutions were improved by BDA, whereas in 1 other instance GSA performed better. Out of the remaining 10 instances in which both methods tied, there are 5 newly proven optimal solutions obtained by BDA.

Both BDA, ACO algorithm, and GSA heuristic were able to reach the same number of workers as the optimal SALBP value in their solutions for the whole solved dataset (Tables 15, 16, and 17). Instances that the proposed BDA outperformed ACO algorithm and GSA heuristic were solved with a reduced number of stations, which indicates a tendency that it is more profitable to accept the SALBP optimal solution as the number of workers, and try to minimise as much as possible the line length (i.e. the number of multi-manned stations). An evidence to support this methodology is that the possibility to also reduce the number of workers in a multi-manned assembly line presented in Section 4.2 was not verified in any instance of the benchmark. Finally, it was verified that the BDA has a tendency in spending the majority of its computational processing time solving SPs in most cases. Besides, Cut1 is less frequently added

Table 16 – Results comparison for medium-size instances between BDA, ACO algorithm, and GSA heuristic.

Prob	CT	NW	BDA								ACO/GSA	
			St	UB	LB	Gap	CPU	SCPU	Cut1	Cut2	HT	Obj	CPU[1]
Tonge	176	2	*	2112	2112	0.0%	16.87	3.86	4668	0	755	2112	14.54
		4	*	2110	2110	0.0%	36.99	30.86	610	5940	1217	2110	27.82
		6	*	2110	2110	0.0%	56.15	50.26	0	7720	1683	2110	41.17
	364	2	*	**1005**	1005	0.0%	19.89	19.57	534	0	104	1007	–
		4	*	**1004**	1004	0.0%	0.95	0.76	0	24	6	1007	–
		6	*	**1004**	1004	0.0%	17.51	16.99	0	162	53	1007	–
	410	2	*	905	905	0.0%	62.74	62.68	35	0	1	905	12.07
		4	*	**903**	903	0.0%	13.94	13.86	0	16	2	904	24.71
		6	*	**903**	903	0.0%	91.80	91.68	0	96	12	904	37.10
	468	2	*	804	804	0.0%	0.18	0.14	0	0	0	804	–
		4	*	**803**	803	0.0%	273.54	273.41	0	48	8	804	–
		6	*	**803**	803	0.0%	161.06	160.93	0	52	2	804	–
	527	2	*	704	704	0.0%	0.18	0.15	4	0	0	704	13.35
		4	*	703	703	0.0%	119.85	119.82	0	57	9	703	26.35
		6	*	703	703	0.0%	575.63	575.57	0	132	32	703	38.75
Arcus1	5048	2	*	1610	1610	0.0%	23.94	5.44	970	0	53	1610	17.95
		4	*	1610	1610	0.0%	153.58	99.78	0	1200	56	1610	34.42
		6	*	1610	1610	0.0%	122.59	92.61	0	1120	65	1610	50.92
	5853	2	*	**1408**	1408	0.0%	8.17	0.34	70	0	8	1410	–
		4	*	**1408**	1408	0.0%	12.25	0.86	0	80	4	1410	–
		6	*	**1408**	1408	0.0%	13.66	0.54	0	20	1	1410	–
	6842	2	*	**1207**	1207	0.0%	3.45	3.06	472	0	34	1208	–
		4	*	**1207**	1207	0.0%	52.21	51.72	0	544	55	1208	–
		6	*	**1207**	1207	0.0%	160.19	159.68	0	656	80	1208	–
	7571	2	*	1106	1106	0.0%	1.18	1.05	18	0	3	1106	17.20
		4	*	1106	1106	0.0%	5.90	5.79	0	48	3	1106	32.57
		6	*	1106	1106	0.0%	38.80	38.66	0	90	26	1106	47.78
	8412	2	*	1006	1006	0.0%	0.22	0.09	0	0	0	1006	–
		4	*	1006	1006	0.0%	0.32	0.22	0	0	0	1006	–
		6	*	1006	1006	0.0%	0.22	0.12	0	0	0	1006	–
	8998	2	*	**905**	905	0.0%	30.60	30.38	12	0	2	906	–
		4	*	**905**	905	0.0%	2.88	2.72	0	12	0	906	–
		6	*	**905**	905	0.0%	0.67	0.50	0	18	3	906	–
	10816	2	*	**804**	804	0.0%	87.83	84.74	485	0	54	805	17.10
		4	*	**804**	804	0.0%	2.46	0.03	0	0	0	805	32.78
		6	*	**804**	804	0.0%	14.17	11.27	0	170	4	805	47.45
Mukhe	176	2	*	2516	2516	0.0%	52.07	12.39	4368	0	392	2516	30.28
		4	I	2513	2512	0.1%	1h	3479.8	20072	13793	2744	2513	60.99
		6	I	2513	2512	0.1%	1h	3478.8	312	11089	453	2513	95.78
	248	2	*	1810	1810	0.0%	29.61	17.84	6750	0	1073	1810	30.71
		4	I	1808	1807	0.1%	1h	3523.2	528	7112	982	1808	63.44
		6	*	1807	1807	0.0%	785.02	78.75	56	868	59	1807	98.48
	351	2	I	1308	1307	0.1%	1h	3495.71	10800	0	550	1308	31.02
		4	I	1307	1306	0.1%	1h	3520.5	3115	4095	332	1307	64.09
		6	*	1306	1306	0.0%	188.23	187.51	0	90	6	1306	99.53

[1]As reported in Kellegöz (2017).

Source: Michels *et al.* (2019).

Table 17 – Results comparison for large-size instances between BDA, ACO algorithm, and GSA heuristic.

Prob	CT	NW	BDA								ACO/GSA	
			St	UB	LB	Gap	CPU	SCPU	Cut1	Cut2	HT	Obj	CPU[1]
Arcus2	5755	2	*	**2714**	2714	0.0%	331.42	303.84	5888	0	554	2716	26.22
		4	*	**2712**	2712	0.0%	1900.1	1880.7	1391	3965	572	2713	48.78
		6	*	**2712**	2712	0.0%	861.55	840.12	0	2236	183	2713	73.00
	8847	2	*	**1811**	1811	0.0%	2593.9	2521.8	25839	0	4116	1812	–
		4	*	**1810**	1810	0.0%	304.24	287.28	55	1562	23	1812	–
		6	*	**1810**	1810	0.0%	12.06	1.53	0	253	0	1812	–
	10027	2	*	**1609**	1609	0.0%	151.24	150.42	2241	0	312	1610	–
		4	*	**1607**	1607	0.0%	1147.9	1134.5	90	2808	240	1610	–
		6	*	**1607**	1607	0.0%	676.12	669.81	0	837	37	1610	–
	10743	2	*	**1508**	1508	0.0%	328.60	303.63	1323	0	34	1509	31.41
		4	*	**1507**	1507	0.0%	610.87	608.03	8	1040	98	1508	61.87
		6	*	**1507**	1507	0.0%	558.84	555.84	0	544	32	1508	92.53
	11378	2	*	**1407**	1407	0.0%	194.89	193.96	904	0	266	1409	–
		4	*	**1406**	1406	0.0%	1298.1	1296.8	1547	1330	388	1409	–
		6	*	**1406**	1406	0.0%	1636.3	1634.4	0	3374	742	1409	–
	17067	2	*	905	905	0.0%	0.26	0.17	5	0	0	905	29.44
		4	*	**904**	904	0.0%	119.42	119.11	0	195	9	905	57.68
		6	*	**904**	904	0.0%	339.91	339.42	0	720	25	905	84.96
Bart2	84	2	I	5127	5126	0.1%	3599.2	2.51	4576	0	14	*5126*	68.00
		4	I	5116	5113	0.1%	3602.3	1839.3	90272	13648	1460	5116	138.64
		6	I	5114	5111	0.1%	3608.8	2603.6	11102	46060	628	5114	224.44
	106	2	I	4121	4020	2.4%	3600.2	2.90	2688	0	186	4121	68.96
		4	I	*4111*	4010	2.5%	3615.9	2175.3	10751	1664	256	4113	145.09
		6	I	*4110*	4010	2.4%	3601.5	3192.3	1896	7200	245	4112	234.68
	170	2	*	2513	2513	0.0%	20.37	2.33	143	0	9	2513	67.18
		4	*	**2507**	2507	0.0%	199.16	196.54	40	24	16	2508	144.08
		6	*	**2506**	2506	0.0%	169.62	168.35	24	16	4	2508	232.94
Bart	403	2	*	1407	1407	0.0%	218.58	217.96	140	0	7	1407	66.10
		4	*	**1404**	1404	0.0%	70.28	70.02	0	5	0	1405	143.39
		6	*	**1404**	1404	0.0%	1004.3	1002.6	25	120	12	1405	228.79
	513	2	*	1106	1106	0.0%	2.69	1.90	0	0	0	1106	66.57
		4	*	**1103**	1103	0.0%	1708.6	1708.2	168	16	17	1104	142.99
		6	*	**1103**	1103	0.0%	3797.8	3794.9	120	300	28	1104	226.17
	805	2	*	704	704	0.0%	75.13	74.83	8	0	0	704	66.55
		4	I	703	702	0.1%	3640.3	3638.3	183	174	0	703	140.12
		6	I	703	702	0.1%	3701.6	3700.9	3	333	0	703	216.74

[1] As reported in Kellegöz (2017).

Source: Michels *et al.* (2019).

than Cut2 when the maximum number of workers is increased and the hash-table is more often consulted for lower values of cycle time.

In order to conduct a feasibility check, the task-worker-station allocation results were tested for all newly found solutions. Task starting and ending times for each worker were examined for consistency regarding station and global cycle times, as well as precedence relation imposed orders. Filling their purpose to validate the proposed BDA's reliability, these tests are

made available along with task-station-worker allocation results in the supporting information files for reproducibility purposes.

4.6 CONCLUSIONS

The Multi-manned Assembly Line Balancing Problem (MALBP) with the objective of minimising the number of workers and stations has been addressed in this study. The existing literature on MALBPs indicated a lack of efficient exact solution methods for these problems, since past mathematical formulations were only able to solve some instances with up to 45 tasks. This chapter's main contribution is solving to optimality MALBPs up to 148 tasks by decomposing the original problem and implementing a Benders' decomposition algorithm employing combinatorial cuts during its execution.

A new Mixed-Integer Linear Programming (MILP) model was developed along with several valid inequalities that work as symmetry break constraints to solve the optimisation problem. This proposed formulation outperforms previously presented monolithic mathematical formulations in terms of solution quality and computational processing time. By analysing the MALBP's structure, it is possible to infer that the problem is divisible hierarchically into a Master Problem (MP) and a Slave Problem (SP), and hence forging a Benders' Decomposition Algorithm (BDA). After adapting several logical cuts inherited from project scheduling problems, the MP solves task-station and worker-station assignment problems, whilst the SP deals with the task-worker scheduling problem for each station, detects infeasibility, and generates combinatorial Benders' cuts to be added into MP as lazy constraints during BDA's execution. The proposed BDA was compared to previously developed methods and was shown to produce improved results while maintaining reasonable CPU time. In total, 42 new optimal solutions were obtained, 2 integer solutions were improved, and 18 previously known solutions were proven optimal out of a dataset with 131 instances.

Allowing multiple workers to perform different tasks simultaneously in the same station is a natural extension of the simpler version of the problem, as well as a notable realistic feature widely employed in industries manufacturing large-size products. Nonetheless, incorporating more practical extensions such as line layout (U-line, parallel stations), product variety (multi and mixed model lines), and zoning restriction in the BDA is a desirable modification. Further research should focus on doing so and, in order to mitigate computational burden, might include balancing and project scheduling heuristics for the master and slave problems, respectively.

5 THE TYPE-2 MULTI-MANNED ASSEMBLY LINE BALANCING PROBLEM

This chapter contains a slightly modified version of the paper Michels *et al.* (2020), which is entitled "*An exact method with decomposition techniques and combinatorial Benders' cuts for the type-2 multi-manned assembly line balancing problem*" and was published in the *Operations Research Perspectives*.

Section 5.1 introduces the type-2 multi-manned assembly line balancing problem (MALBP-2). Section 5.2 conducts a thorough literature review on related multi-manned problems. Section 5.3 describes the studied problem, showing the main advantages of adopting multi-manned stations and its specific assumptions are compared to the ones in literature. Section 5.4 presents a new MILP model with valid inequalities and upper bounds for the MALBP-2. Section 5.5 gives an overview of the proposed solution method and how it is executed for the studied problem. Section 5.6 validates the results of the proposed method by optimally solving many MALBP-2 instances in the benchmark dataset and yielding improved primal bounds for an industrial case study. Final considerations of this chapter are presented in Section 5.7.

Abstract from Michels *et al.* (2020): *"Multi-manned assembly lines are widely applied to manufacturing industries that produce large-size products and are concerned with high levels of productivity. Such lines are commonly found in automotive industries, where different tasks are simultaneously performed by more than one worker on the same product in multi-operated stations, giving rise to a class of balancing problem that aims to minimize the line's cycle time. This clear practical application had made the type-2 multi-manned assembly line balancing problem to be explored in the past. However, only few small-size instances could be solved by preceding exact solution approaches, whereas large and real-life cases still lack optimality proofs since they were tackled by heuristics. In this work, a new Mixed-Integer Linear Programming model is presented and its modeling decisions discussed. Moreover, an innovative exact solution procedure employing a combination of decomposition techniques and combinatorial Benders' cuts is presented to solve large and real-life instances optimally. Tests on an extended literature dataset and a real-life assembly plant case study have demonstrated that the proposed algorithm outperforms previously developed methods in terms of solution quality by an ample margin in efficiency gains. Synergies between the algorithm's components are also revealed. Finally, the proposed exact method has been able to yield 60 optimal results out of a 108-instance dataset, with the remaining 48 solutions presenting a small integer gap (less than 2%)."*

5.1 INTRODUCTION

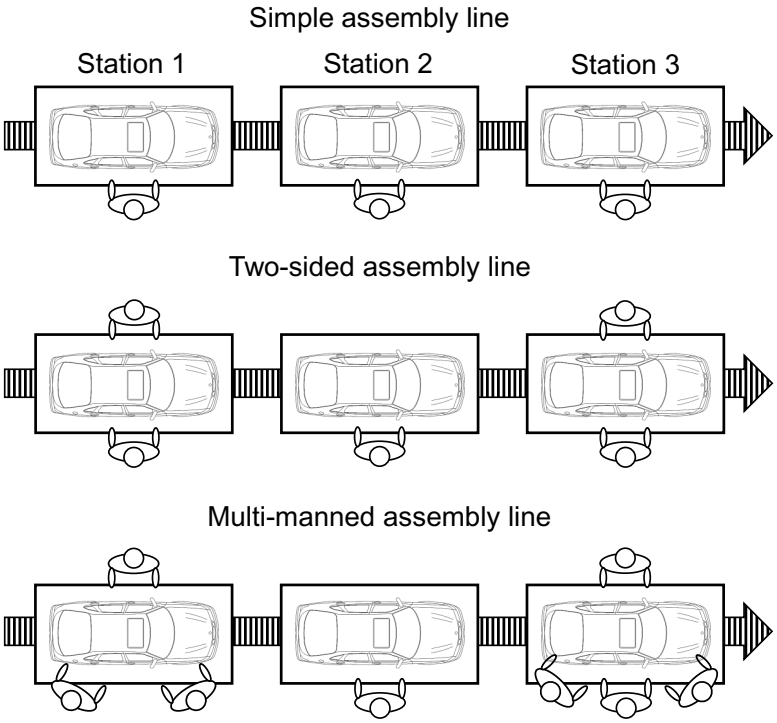
Assembly lines are production systems usually applied to large-scale industries of similar and standardized products. They are frequently built to fit homogeneous products and enable their mass production, hence generally designed as flow-shop layouts. In the academic research, the Assembly Line Balancing Problem (ALBP) is widely discussed in the literature, which is the main combinatorial problem that assembly lines have created Battaïa and Dolgui (2013).

In its simplest form, the Simple Assembly Line Balancing Problem (SALBP) consists in assigning a list of tasks subjected to a precedence graph to stations, whilst considering various simplification hypotheses described by Baybars (1986). One of them is to solely allow one worker to operate each station. Furthermore, a serial, straight line organization is imposed to the stations. In turn, such line produces a single product model. The most common goals in a SALBP are (i) minimizing the number of stations (SALBP-1) or (ii) minimizing the cycle time (SALBP-2) and important contributions to its literature was surveyed by Scholl and Becker (2006). Among those, algorithmic solution methods were proposed and related to practical applications: developed by Scholl and Klein (1997), SALOME is an effective procedure based on a bidirectional branch-and-bound, while Bautista and Pereira (2009) proposed a dynamic programming approach, which were followed by Sewell and Jacobson (2012) and their branch, bound, and remember algorithm, as well as a multi-Hoffmann heuristic with enhanced properties proposed by Sternatz (2014). For an overview and improved techniques for SALBP-1, it is possible to refer to Pape (2015).

Notwithstanding, by revisiting the “only one worker per station" hypothesis applied to SALBPs, one can verify that it often is not a practical limitation on, for instance, manufacturing lines found in automotive factories. These industries regularly produce cars, buses, and trucks, which are large-size products, broadly speaking. Because the physical area occupied by stations are quite large compared to an operator’s working space, multiple workers can simultaneously perform tasks on the same product. Accordingly, more generalized extensions of the SALBP come to light as natural extensions: the Multi-manned Assembly Line Balancing Problem (MALBP) and the Two-sided Assembly Line Balancing Problem (TALBP). Both problems, along with an assortment of other practical extensions, are surveyed by Becker and Scholl (2006).

These above mentioned lines are illustrated in Figure 19. It exhibits examples of assembly lines composed of three stations each: simple, two-sided, and multi-manned assembly lines are respectively depicted. As observed, both two-sided and multi-manned assembly lines admit multiple workers in each station (first and third stations), with workers simultaneously performing tasks on the same product. Notice, however, that MALBPs are more flexible than TALBPs in terms of the quantity of workers and their positioning. The main difference is that two-sided lines allow at most one operator in each side of the station (right and left), whereas multi-manned lines have their maximum number of workers set depending on product's attributes. These can be seen as size, structure, and tasks' specificities. The last divergence is that TALBPs may have to deal with tasks exclusively performed on the right or left side of the product. Both of these problems can also be solved considering analogous goal functions studied for SALBP (BECKER; SCHOLL, 2006).

Figure 19 – Layout examples of simple, two-sided, and multi-manned assembly lines.



Source: Michels *et al.* (2020).

This work focuses on the MALBP-2 variant, namely cycle time minimization. A previous MALBP-1 specialized algorithm (MICHELS *et al.*, 2019), when adapted, was not capable to satisfactorily solve the proposed problem, not even generating feasible solutions for most large-size instances, and the best-known meta-heuristic for the MALBP-2 (ROSHANI; GIGLIO, 2017) produces several sub-optimal solutions. Therefore, the main contribution to

the state-of-the-art herein proposed comes from the implementation of an innovative exact algorithm to optimally solve the MALBP-2 dataset. Some relevant simplification hypotheses from SALBP-2 mentioned in Baybars (1986) are kept, viz. a serial, straight line is considered and such line produces a single, unique model. The remaining of the chapter is organized as follows. In Section 5.2, the relevant literature is introduced, focusing on specific objective functions, extensions, and methodologies used by each author. In order to define the problem, Section 5.3 precisely explains the characteristics of a type-2 MALBP, demonstrating advantages associated to obtainable production rate improvement and line length reduction when using a multi-manned configuration. Section 5.4 presents the MILP model, valid inequalities, upper and lower bounds, and a brief discussion on modeling decision. Section 5.5 describes the development and applications of the proposed Benders' decomposition algorithm and combinatorial cuts, as well as how an initial solution is constructed for the problem at hand. Computational results retrieved from a benchmark dataset and a real-life assembly plant case study are presented and discussed in Section 5.6. Finally, in Section 5.7, a summary of the concluding remarks and further research directions are described.

5.2 LITERATURE REVIEW

As a consequence of the intense use of multi-operated stations in industrial environments, several studies regarding MALBPs and TALBPs have been elaborate in the literature. To the best of the authors' knowledge, Akagi *et al.* (1983) is the first study to take on the problem of allowing more than one worker per station in assembly lines, while attaining good production rates. For that, an approach denominated Parallel Assignment Method (PAM) was developed.

Much later, a heuristic method to address such problem was introduced by Dimitriadis (2006), which was based on altering a procedure previously created by Hoffmann (1963). Given a fixed cycle time, it has been shown that the heuristic was useful in enhancing stations physical utilization. The objective of minimizing the total number of workers and stations was considered, which turns out to be the most customary goal function exploited in subsequent works since then.

Following those pioneer publications, the attention on MALBPs have been growing in the last ten years. Becker and Scholl (2009) proposed the Assembly Line Balancing Problem with Variable Workplaces (VWALBP), in which a cycle time is given and working areas are minimized. Here the product is fragmented into mounting positions, while imposing in each multi-manned station that only one worker is capable of assembling them. A Mixed-Integer Linear

Programming (MILP) formulation is modeled in conjunction with lower bounding techniques. Besides, a branch-and-bound algorithm based on SALOME – called VWSolver – is developed to solve large instances. In sequence, a novel efficient algorithm was implemented by Kellegöz and Toklu (2012) to tackle ALBPs with multi-manned stations (Jumper), which is also based on branch-and-bound procedures. Jumper outperforms VWSolver in both computational processing times and quality of feasible solutions.

Fattahi and Roshani (2011) were the first to propose a mathematical formulation that simultaneously minimizes the total number of workers and stations in a MALBP, defining the type-1 variant. In the optimization procedure, minimizing the number of workers is the primary objective and minimizing the number of stations is the secondary one. Keeping the remaining SALBP hypotheses, their model failed in solving medium and large cases, but was able to solve small-size test problems in an acceptable time limit. To settle this issue, they have developed an Ant Colony Optimization (ACO) algorithm. The ACO algorithm could find the same optimal solutions for small-size problems in a much reduced computational time, as well as feasible and near-optimal solutions for many medium and large instances. Later, Kellegöz (2017) has created a better MALBP-1 mathematical model proposed by Fattahi and Roshani (2011). Additionally, a Gantt-based heuristic was developed within a Simulated Annealing (SA) framework. This procedure is able to solve medium and large-size instances, finding improved feasible solutions to a large number of instances in the tested benchmark. Therefore, it has been concluded that the GSA algorithm outperforms the ACO algorithm given by Fattahi and Roshani (2011).

The examination of assembly lines with multi-operated stations regarding cost-oriented problem instances has been conducted by Kazemi and Sedighi (2013) and Michels *et al.* (2018b) on real-size case studies. The former presents a heuristic method based on Genetic Algorithm (GA) that takes into account the objective of minimizing costs per production unit, whereas the latter analyzes a robotic assembly line that considers the employment of multiple robotic workers per station, conceiving robots, facilities, and tool prices. Hence, an MILP model to minimize implementation costs for the line design is developed.

A multi-objective function to address MALBPs is firstly proposed by Roshani *et al.* (2013). By maximizing line efficiency and minimizing the line length and smoothness index, a mathematical model is formulated. Furthermore, an improved SA algorithm has been developed to tackle the problem. Meanwhile, a constructive heuristic based on priority rules followed by an improvement procedure based on GA has been presented by Kellegöz and Toklu (2015).

Computational experiments have been conducted on MALBP instances to minimize the total number of workers in the line. At this point, Yilmaz and Yilmaz (2015) have created a new mathematical formulation, aiming at the minimization of total number of workers, stations, as well as workload difference between workers. Subsequently, Yilmaz and Yilmaz (2016a) also studied the impacts of equipment needs and skilled workers on MALBPs. In order to solve the problem, a heuristic procedure was proposed.

For a given fixed number of stations, Roshani and Giglio (2017) approached the MALBP by attempting to minimize the cycle time and the number of workers in a line as the primary and secondary objective, respectively. This strategy is further discussed in Section 5.4.3, as it neglects fundamental MALBP hypotheses regarding the higher importance of workers over stations, flexibility gains over SALBP solutions, and conditions found in practice. Alongside the MILP model, two meta-heuristics, the indirect and direct SA algorithm (ISA and DSA, respectively), were implemented. In both terms of solution quality and computational time. Therefore, the DSA showed to be more efficient, with a better performance in solving the problem. Moreover, another SA algorithm was developed by Roshani and Nezami (2017), this one undertakes the mixed-model variant of MALBP-1.

More recently, both Naderi *et al.* (2019) and Michels *et al.* (2019) developed and applied Benders' decomposition algorithms (BDA) (BENDERS, 1962) with combinatorial Benders' cuts (CBC) (CODATO; FISCHETTI, 2006) to the MALBP-1 variant, reaching outstanding results. The former is able to solve a realistic five-sided MALBP with moving workers and limited workspace, whereas the latter solves various medium and large-size MALBP-1 instances optimally, proving the optimality of some solutions reported by Kellegöz (2017) and improving many others in terms of solution quality. The Benders' decomposition method aspires to reformulate an original and complete monolithic model into a master problem (MP) and slave (or sub) problems (SP), transforming them into two hierarchical problems. This partition aims at removing the burden of several variables and restrictions from the MP, transferring part of the load to one or more SPs, which are then solved individually. By iteratively working between the MP and SPs, solutions can be found and evaluated much faster: values for key variables are fixed in a given solution of the MP, then used in the SPs to define the remaining ones. Each MP candidate solution is generally decomposable into multiple smaller problems, which is translated into a set of SPs that are solved individually, supposedly reducing the computational burden when compared to straightforwardly solving the monolithic problem. Each SP yields a solution that

can be used to inform the MP of infeasible combinations: these are known as CBC. The MP with these added cutting planes (extra CBC restrictions) is then solved and the procedure iterates with the next result. Specifically, this decomposition method strips off complicated variables from the MP, then repeatedly corrects misled solutions by solving the SPs (MICHELS *et al.*, 2019). These corrections are made by finally applying CBC to the MP (CODATO; FISCHETTI, 2006), pointing out infeasibilities caused by parts that were omitted in it. As BDAs are highly dependent on the problem being dealt with, straightforward adaptations of previous algorithms rarely work as well as intended. Thus, authors generally develop specific algorithms to their problems, incorporating as much particular information as possible from the problem under study.

Rahmaniani *et al.* (2017) surveys and summarizes the use of BDAs in the literature, where only one work concerning a transfer line balancing problem is listed in the review (OSMAN; BAKI, 2014). Nonetheless, it was verified by further investigation that, besides the aforementioned works, various real-world problems have been approached by using BDAs. However, related research was scarcely developed for assembly line balancing problems. In Hazir and Dolgui (2013) and Hazir and Dolgui (2015), different layouts (straight lines and U-lines) are considered under uncertainty. A robust optimization model and a specialized algorithm are formulated for each case. Lastly, Akpinar *et al.* (2017) task-dependent set-up times are taken into account. In this problem, task assignment and task sequencing decisions can be interpreted as hierarchical problems.

Finally, other recent works worth mentioning on MALBP variants are: Sahin and Kellegöz (2019a), which considered resource investment and developed a hybrid heuristic based on Particle Swarm Optimization (PSO) to tackle the problem; Sahin and Kellegöz (2019b) took into account the possibility of walking workers (SIKORA *et al.*, 2017a) in the line and solved large instances with a reduced integer gap by applying an Electromagnetic Field Optimization (EFO) algorithm to the problem; Lopes *et al.* (2019) pointed out inconsistencies in a previously published paper (CHEN, 2017) and proposes a model based hierarchical decomposition procedure to minimize the number of stations after the task-worker assignment solution has been decided in a multi-manned assembly line; Lopes *et al.* (2020) brought more flexibility to multi-manned lines by explicitly considering continuous paced line control, relaxing the limitation of fixed, discrete, and restrictive frontiers between stations. Their model, heuristic procedure, and algorithmic lower bounds accommodate significantly shorter line lengths; Yilmaz and Yilmaz (2020) proposed

a mathematical model to consider assignment restrictions, i.e. positive and negative zoning, distance, station, and synchronous task restrictions. A tabu-search algorithm is also developed to effectively solve the problem; and Yadav *et al.* (2020) improved worker and station efficiencies of a real-life automotive plant by reconfiguring the assembly line under study.

In summary, none of the reviewed works but Roshani and Giglio (2017) considered the type-2 MALBP. Nevertheless, by using SA algorithms that would not guarantee optimality, they obtained sub-optimal solutions with lower levels of efficiency than its SALBP-2 counterpart. In order to bridge this gap, a new mathematical formulation with different modeling decisions is developed with valid inequalities to address the problem. Furthermore, a solution method for the problem under study is proposed. It combines the decomposition technique introduced by Lopes *et al.* (2019) to find an initial assignment with a BDA as the strategy to find and prove optimal solutions for MALBP-2 instances. Larger benchmark instances can be solved to optimality by applying CBC (BENDERS, 1962; CODATO; FISCHETTI, 2006) as lazy constraints while the algorithm is executing. Unlike classical Benders' decompositions with linear SPs (BENDERS, 1962), the proposed algorithm presents integer slave problems (CODATO; FISCHETTI, 2006), which are intended for feasibility seeking, as in Michels *et al.* (2019) and Naderi *et al.* (2019). Besides Roshani and Giglio (2017) being the only work concerning MALBP-2, they likewise supply an comprehensive dataset to validate both the proposed model and algorithm. Therefore, these instances are provided as a benchmark for this work and the solution method herein proposed, which defines the MALBP-2 and focus on minimizing the cycle time as the primary objective and the number of stations as the secondary given a fixed number of worker. In this way, objective function results can be directly compared to evaluate the algorithm's performance.

5.3 PROBLEM STATEMENT

As aforementioned, multi-manned stations from assembly lines examined in this study are employed to large-scale production of a single model of large products. Such stations are sequentially positioned in a straight, serial line. Work-pieces can just be processed one at a time in each station. These pieces move forward between stations within a cycle time (CT) to be optimized (minimized), whilst their transfer times between stations are neglected. As a single product is produced in the line, its pace – and consequently its production rate – is entirely determined by the most loaded station (BAYBARS, 1986).

A set of indivisible tasks T must be performed in order to assemble any product. Since

precedence relations must be respected, they cannot be executed in an arbitrary order. Tasks have a deterministic processing duration time (D_t) to be finished. Therefore, the sum of task processing times assigned to the same worker must not surpass the cycle time limit, which is imposed by the most loaded worker. However, the possibility of parallel work within each station makes it essential to schedule tasks in such a manner that precedence restrictions are still respected.

In Table 18, an illustrative instance is sampled to depict differences between SALBP and MALBP possible results. This illustrative instance will be used as a numerical example throughout the chapter to explain the proposed method. For the considered precedence graph, task index numbers are ordered on the first line, their respective durations are represented by the value just below it, and their direct predecessor are given in the last line. Additionally, optimal solutions for both SALBP (Figure 20(a)) and MALBP (Figure 20(b)) versions of this instance are shown in the format of Gantt diagrams, with a defined maximum number of workers in the line to be five ($Nmax = 5$). Differently from multi-manned lines, the total number of workers and stations must be the same for SALBPs. In the former, notwithstanding, each work-piece is allowed to be engaged by more than one worker employed in the same station simultaneously. Due to product sizes, the maximum number of workers that a station can suit to perform different tasks concomitantly (NW) may vary. In this example, $NW = 3$.

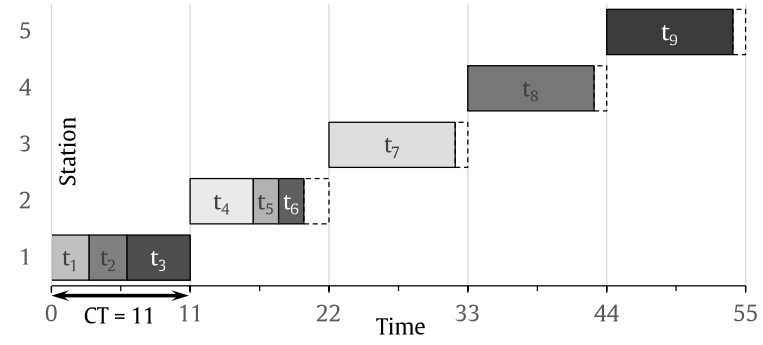
Table 18 – Task duration times and precedence relations for the illustrative instance.

Task	1	2	3	4	5	6	7	8	9
Duration	3	3	5	5	2	2	10	10	10
Precedence	–	–	1,2	1,2	3,4	3,4	5	5,6	6

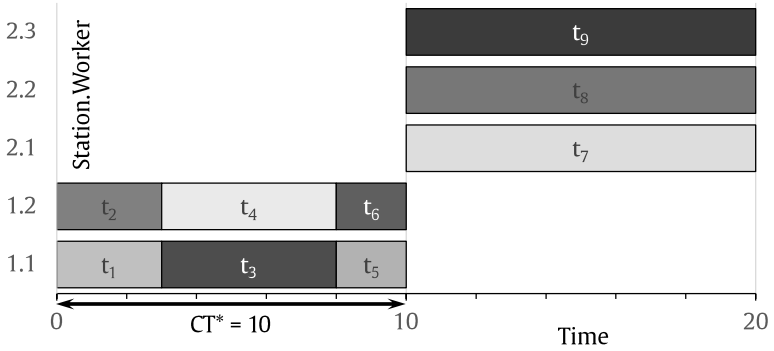
Source: Michels *et al.* (2020).

The SALBP's solution requires 5 workers (represented by bars) assigned to 5 stations (on the y-axis) to deliver a production rate (with times on x-axis) of one product unit each 11 time units (i.e. $CT_{SALBP} = 11$) in its optimal configuration. Represented by dashed blank spaces in each station, idle times total 5 time units along the line, which translates to approximately 9.09% of the line's available task performing time. On the other hand, by permitting simultaneous operations within stations by different workers, the MALBP optimal solution could, at the same time, reduce the line length from 5 to 2 stations and improve the production rate to a cycle time of 10 time units ($CT_{MALBP} = 10$) instead of 11 to perform the same task set. Hence, an advantage of multi-manned lines over simple ones was demonstrated: the additional flexibility allowed idle time to be reduced to zero. Naturally, these improvements depend on the instance's

Figure 20 – Gantt diagram representations of SALBP-2 and MALBP-2 optimal solutions for the illustrative instance.



(a) SALBP-2 solution



(b) MALBP-2 solution

Source: Michels *et al.* (2020)

parameters, but it is clear that efficiency gains can occur at no extra or even reduced cost when multiple workers are allowed to perform different tasks at the same time within a station.

Nevertheless, tasks must still be scheduled within each station to ensure that precedence relations are respected. These task-scheduling requirements imply on a computational burden to solve the problem, which inevitably comes along as a drawback by allowing such flexibility to acquire the presented viable advantage. Notice that, into the same station, at most one task can be executed by each worker at a given time, whilst cooperation between workers is forbidden, which means that common tasks are not considered (YAZGAN *et al.*, 2011; SIKORA *et al.*, 2017a). Moreover, positioning and zoning restrictions do not constrain task assignments (BARTHOLDI, 1993; BECKER; SCHOLL, 2009), and interferences between workers do not happen during the assemblage process (LOPES *et al.*, 2017) as long as precedence relations are respected. Workers are homogeneous (MOREIRA *et al.*, 2015), meaning all of them have the same regular capability and can perform any task with the same specific time duration required for its execution. As

stated, even though different workers can perform different tasks synchronously, all precedence relations enforced by the precedence graph still must be satisfied. Consequently, for each station a task scheduling problem originates, with possible waiting (idle) times for workers between or before the performance of tasks. These must be taken into account, thus making MALBPs more complex and difficult to solve than SALBPs (FATTAHI; ROSHANI, 2011).

Due to high line utilization associated to assembly lines, it is assumed that balancing decisions are a long-term plan for the studied problem, which implies in a desire to maximize productivity with the available resources. In this situation, the number of workers that a company can employ is the main fixed resource, as they represent costs related to wages, equipment, labor regulations, among others. Stations and facilities, on the other hand, are a one-time investment that is relatively much cheaper when compared to workers, but should be considered nonetheless (FATTAHI; ROSHANI, 2011; KELLEGÖZ, 2017; MICHELS *et al.*, 2019). Hence, the mathematical model presented in Section 5.4 prioritizes productivity maximization in its objective function, i.e. minimizing cycle time, accompanied by a secondary objective of reducing facility costs, i.e. minimizing the total number of stations and, accordingly, the line length.

Ultimately, solutions that are deemed optimal for SALBPs are necessarily feasible configurations for MALBPs. It follows that the optimal cycle time solution given a fixed number of workers (and stations) achieved by a SALBP can be viewed as an upper bound for its MALBP counterpart, since simple lines are less flexible and restrict the use of workers to only one in each station for any given solution. Correspondingly, the maximum number of stations (NS) in a MALBP can be set to one unit lesser than the optimal solution found by its simpler counterpart. This adopted upper bound is a reasonable measure: as the MALBP-2's goal is to minimize both the cycle time and the total number of stations, with a much lower importance in the latter, the minimal marginal improvement taken from a SALBP solution is keeping the same productivity level while reducing the line length in at least one station. This has been similarly demonstrated by Michels *et al.* (2019) for the MALBP-1 variant. For the illustrative example presented in Figures 20(a) and 20(b), a MALBP-2 instance starts with an upper bound for the number of stations equaling four ($NS = 4$). It is explained by the fact that, if the model is incapable of reducing at least one station in the previous solution (accusing the problem to be infeasible), then it is concluded that having the flexibility to allow multiple worker per station does not help in obtaining efficiency improvements at no additional costs, meaning that both versions (SALBP and MALBP) have the same optimal solution in terms of objective value. This reasoning is

further discussed in Section 5.4.3.

5.4 MILP MODEL

In this section, the developed Mixed-Integer Linear Programming (MILP) model is presented. It represents the type-2 Multi-manned Assembly Line Balancing Problem (MALBP-2), taking into account the characteristics recognized in Section 5.3. Section 5.4.1 outlines the main model to express the problem and Section 5.4.2 shows the implemented constraints that strengthens the problem's linear relaxation (valid inequalities). Furthermore, modeling decisions concerning the objective function and parameter values are discussed in Section 5.4.3. Tables 19 and 20 display the applied terminology to describe parameters, sets, and variables used in the formulation. Variables are created by the model depending on its sets, as detailed in Table 20.

Table 19 – Definition of MALBP-2 parameters and sets.

Parameter	Description
NT	Number of tasks
NS	Upper limit on the number of stations
NW	Upper limit on the number of workers per station
$Nmax$	Upper limit on the number of workers in the line
D_t	Duration [time units] of task t, always a natural number
B	A big positive number
Set	**Description**
T	Tasks set t; $T = \{1, 2,..., t,..., NT\}$
S	Stations set s; $S = \{1, 2,..., s,..., NS\}$
W	Workers set w; $W = \{1, 2,..., w,..., NW\}$
TS	Task-Station tuple
TW	Task-Worker tuple
WS	Worker-Station tuple
TWS	Task-Worker-Station tuple
P	Precedence relations between tasks (t_i,t_j): $t_i \prec t_j$

Source: Michels *et al.* (2020).

Table 20 – Definition of MALBP-2 variables.

Variable	Set	Domain	Description
CT	–	$\mathbb{R}_+$	Cycle time [time units]
$X_{t,s}$	$(t,s) \in TS$	{0,1}	Task assigned to station: 1 if task t is assigned to station s
$Y_{w,s}$	$(w,s) \in WS$	{0,1}	Worker assigned station: 1 if worker w is hired in station s
$W_{t,w}$	$(t,w) \in TW$	{0,1}	Task assigned to worker: 1 if task t is executed by worker w
Z_s	$s \in S$	{0,1}	Open station: 1 if station s is used
F_{t_i,t_j}	$t_i, t_j \in T \mid t_i \neq t_j$	{0,1}	Following: 1 if task t_j follows task t_i
ST_t	$t \in T$	$\mathbb{R}_+$	Start time: task t starts to be performed at this time

Source: Michels *et al.* (2020).

5.4.1 Main model

The formal mathematical definition of the MALBP-2 is given by Expressions 60 to 72:

$$\text{minimize: } \underbrace{CT}_{cycle\ time} + \underbrace{\frac{1}{NT+1} \cdot \sum_{s \in S} Z_s}_{number\ of\ stations} \tag{60}$$

$$\sum_{s \in S} X_{t,s} = 1 \qquad \forall\, t \ \in \ T \tag{61}$$

$$\sum_{w \in W} W_{t,w} = 1 \qquad \forall\, t \ \in \ T \tag{62}$$

$$Y_{w,s} \geq X_{t,s} + W_{t,w} - 1 \qquad \forall\, (t,s) \ \in \ TS,\ (t,w) \ \in \ TW \tag{63}$$

$$Nmax \geq \sum_{(w,s) \in WS} Y_{w,s} \tag{64}$$

$$F_{t_i,t_j} = 1 \qquad \forall\, (t_i,t_j) \ \in \ P \tag{65}$$

$$F_{t_i,t_j} \geq X_{t_j,s} - \sum_{sk \in S | sk \geq s} X_{t_i,sk} \qquad \forall\, t_i, t_j \ \in \ T,\ s \ \in \ S \mid t_i \neq t_j \tag{66}$$

$$F_{t_i,t_j} + F_{t_j,t_i} \geq X_{t_i,s} + X_{t_j,s} + W_{t_i,w} + W_{t_j,w} - 3 \qquad \forall\, (t_i,w,s), (t_j,w,s) \ \in \ TWS \mid t_i \neq t_j \tag{67}$$

$$ST_t \geq CT \cdot (s-1) - B \cdot (1 - X_{t,s}) \qquad \forall\, (t,s) \ \in \ TS \tag{68}$$

$$ST_t + D_t \leq CT \cdot s + B \cdot (1 - X_{t,s}) \qquad \forall\, (t,s) \ \in \ TS \tag{69}$$

$$ST_{t_j} \geq ST_{t_i} + D_{t_i} - B \cdot (1 - F_{t_i,t_j}) \qquad \forall\, t_i, t_j \ \in \ T \mid t_i \neq t_j \tag{70}$$

$$Z_s \geq X_{t,s} \qquad \forall\, (t,s) \ \in \ TS \tag{71}$$

$$Z_s \geq Y_{w,s} \qquad \forall\, (w,s) \ \in \ WS \tag{72}$$

Expression 60 states the objective function considered for this problem, it is akin to the one used in Roshani and Giglio (2017). The first portion in the expression is the same, it corresponds to the line's cycle time. The second component of the objective function, however, represents the total number of stations opened in the line instead of the total number of workers, along with its weighted cost ($\frac{1}{NT+1}$). This crucial modification is further detailed in Section 5.4.3. Notice that, in this manner, the primary objective is to minimize the cycle time: there is a clear hierarchical order of importance between cycle time and number of stations, so that a cycle time unit is at least $1 + NT$ times costlier than a station, enforcing a disadvantageous trade-off between them. It is also important to mention that cycle time values are always integer due to the nature of task processing times: the ALBP benchmark only contains instances with integer task durations and, in any case, one can simply multiply the actual task processing times (given in seconds or minutes) of all tasks by an arbitrary large number in order to obtain integer durations, which is a common practice in the surveyed literature (Section 5.2).

Occurrence is stated by Constraints 61, they force each task to be assigned to a station exactly once. Analogously, Constraints 62 ensure each task to be exclusively executed by a specific worker. Constraints 63 impute appropriate values to $Y_{w,s}$ variables: whenever a task t is assigned to station s and such task t is also performed by worker w, then it is conceivable to infer that worker w from station s is employed for such tasks. Finally, Constraints 64 state that the total number of workers in the line is limited by the maximum number of available workers ($Nmax$).

The precedence relations and scheduling between tasks are satisfied by Constraints 65–70, which use the same reasoning recently presented by Michels *et al.* (2019). Task following (F_{t_i,t_j}) and starting time (ST_t) variables define if and when each task can or must start. It is mandatory for a given task t_j to follow task t_i (Constraints 65) if such pair of tasks (t_i,t_j) is included in P (the precedence set). The variables F_{t_i,t_j} exclude situations in which $t_i = t_j$, since a task cannot follow itself. In order to properly determine following variables based on task assignment and worker use, Constraints 66 and 67 work as logical ties. Constraints 66 decide whether or not t_j follows t_i between stations: task t_i follows task t_j whenever t_i is not assigned to the same station that t_j is or in any station after that. In the same station, however, one of them must follow the other if both tasks t_i and t_j are performed by the same worker (Constraints 67). Introducing a sufficiently large natural number B (whose value is stated in Section 5.4.3), the starting time of a task t must be superior to a minimum value regarding the station s in which it

is performed (Constraints 68), as well as inferior to its upper maximum limit (Constraints 69). Lastly, Constraints 70 state that all tasks t_i that precede t_j must be finished before t_j can start.

Finally, the combination of Constraints 71 and 72 along with the objective function assists in a logical searching process, as they forbid the existence of unoccupied or unproductive stations: they respectively state that a station must be opened for a task or a worker to be assigned there.

5.4.2 Valid inequalities

The model 60–72 represents MALBP-2. Nonetheless, in order to save some time by not visiting symmetric solutions, some ordering symmetry breaks have to be implemented into the model (WALSH, 2006). The problem's linear relaxation is strengthened and the issue found by Yilmaz and Yilmaz (2016b) in a previous paper is prevented by them. In Fattahi and Roshani (2011), objective function values would be correct, but inconsistent solutions were found because an arbitrary opening order of stations was permitted. Therefore, a station can only be opened if a previous one is already opened (Constraints 73) and a worker can only be used if a previous one is already in use (Constraints 74). They break the symmetry between stations and workers, respectively, avoiding equivalent solutions to be taken into account by the model in respect of objective function values, thus shrinking the search-space and leading to tighter bounds. These ordering constraints were also used by Kellegöz (2017), Michels *et al.* (2019), and Naderi *et al.* (2019) in their MALBP-1 mathematical models, provided that Constraints 73 and 74 are just applied to the model from the second station/worker onwards.

$$Z_s \leq Z_{s-1} \qquad \forall\, s \;\in\; S \mid s > 1 \tag{73}$$

$$Y_{w,s} \leq Y_{w-1,s} \qquad \forall\, (w,s) \;\in\; WS \mid w > 1 \tag{74}$$

Henceforth, the MILP formulation defined by 60–74 is referred to as PF (proposed formulation).

5.4.3 Upper and lower bound values for CT

This section explores a crucial modeling decision. It concerns the definition of an upper bound value applied to CT and its role in the objective function. It is necessary to

remind that, by hypothesis, Baybars (1986) define the SALBP to be a very constrained problem, restricted by several simplification assumptions. One of them being that each station can only be operated by one worker. Thus, by fixing the number of stations in a type-2 SALBP, one is, in practice, minimizing the line's cycle time while maintaining the number of workers fixed as a consequence. It makes sense; tasks are actually performed by workers rather than by physical stations. Nevertheless, more than one worker can be engaged in each station once this hypothesis is relaxed. It follows that the number of workers and stations must be expressly disassociated, allowing more flexible and efficient manufacturing configurations. Moreover, it has been repeatedly stated that weights can be attributed to workers and stations based on the explicit economical importance of each resource, in which fixed monthly costs (cumulative salaries) are generally much costlier than the one-time expense of physical parts of a station (FATTAHI; ROSHANI, 2011; KELLEGÖZ, 2017; NADERI *et al.*, 2019; MICHELS *et al.*, 2019) in MALBP-1 works.

For a MALBP-2, Roshani and Giglio (2017) have chosen to minimize the cycle time as the primary goal and the number of workers along the line as the secondary, all that whilst fixing the total number of stations. As a consequence, their model tends to fit as many workers as possible in the stations in order to achieve better levels of productivity. Thus, the results reported in their paper show solutions in which a SALBP-2 configuration with the same number of workers could be more efficient in terms of cycle time than the one optimized for MALBP-2. Considering that the primary objective is to minimize CT, optimized MALBP solutions should have been at least equivalent to its SALBP counterpart (MICHELS *et al.*, 2019), but never worse. Taking that issue into consideration, it has been decided that the proposed formulation would fix the number of workers along the line ($Nmax$) and minimize the total number of stations as the secondary objective. The reasoning behind it is that, in this way, the problem is closer to its SALBP-2 relative (minimize cycle time) and still accepts the concept of previously developed MALBP-1 works, in which the number of workers is the resource that should be prioritize over the number of stations. This approach is also in accordance with what can be found in reality: it does not exaggerate labor costs to achieve better production rates. In fact, overall costs must be lower than (or at least equal to) its less flexible counterpart, a characteristic that cannot be found in Roshani and Giglio (2017).

With the number of workers as the fixed resource, it is expected that the proposed MALBP-2 formulation cycle time is at least as good as its SALBP-2 version with the same given

number of workers. That said, Constraints 75 are included in the model to represent an upper bound for the cycle time value in a MALBP-2, in which CT_{SALBP} is a newly introduced parameter for the optimized SALBP-2 variation of the same data. On the other hand, Constraints 76 and 77 are trivial lower bounds for the cycle time and the total number of stations, respectively. Moreover, $CT_{SALBP} \cdot NS$ is sufficiently large to assume the B role in the PF.

$$CT \leq CT_{SALBP} \tag{75}$$

$$CT \geq \left\lceil \frac{\sum_{t \in T} D_t}{Nmax} \right\rceil \tag{76}$$

$$\sum_{s \in S} Z_s \geq \left\lceil \frac{Nmax}{NW} \right\rceil \tag{77}$$

5.5 SOLUTION METHOD

The Benders' decomposition algorithm (Section 5.5.2) – hereafter referred to as BDA – and its initial solution procedure (Section 5.5.1) developed for the MALBP-2 are herein presented. Section 5.5.2.1 presents the master problem (MP), whilst Section 5.5.2.2 is dedicated to the slave problem (SP). In the proposed implementation, the former is associated to decisions concerning task and worker assignments to stations (high-level problem) and the latter is related to feasibility tests of task assignments and scheduling to workers in each station (lower-level problem). However, since Benders' decompositions are known to have a slow convergence process (MAGNANTI; WONG, 1981), the algorithm may go through enhancement modifications to accelerate its operation. Those are explicitly pointed out along with the initial solution, MP, and SP descriptions.

5.5.1 Initial solution decomposition

An initial feasible solution for the MALBP-2 can be obtained by following the routine presented in Sections 5.5.1.1 and 5.5.1.2. Afterwards, the task-worker-station allocation and scheduling found by this procedure feeds the Benders' decomposition algorithm, which is, in turn, presented in Section 5.5.2. The illustrative instance presented in Table 18 is solved step-by-step as the proposed solution method stages are presented.

5.5.1.1 Solving the SALBP counterpart

In the proposed solution method, the first step to solve a MALBP-2 instance is to solve its simpler counterpart, the SALBP-2. For that, the well-known exact algorithm SALOME (SCHOLL; BECKER, 2006) is used to find the optimal or an integer solution for a given case. By applying this step to the illustrative instance (Table 18), the solution previously presented in Figure 20(a) can be obtained, with 5 workers placed into 5 stations, and a cycle time of 11 time units.

5.5.1.2 Minimizing stations

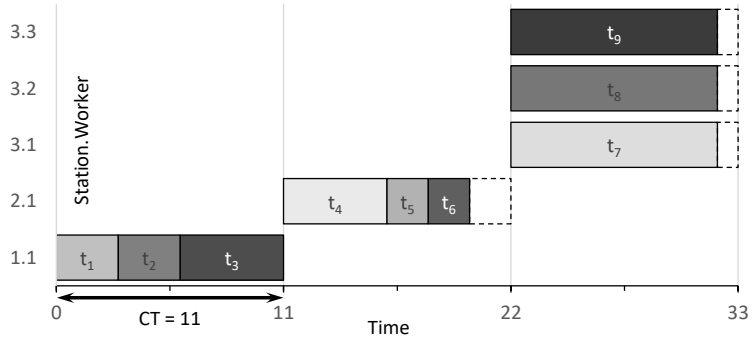
Since SALBPs assume that there is always only one worker in each station, once with a feasible task-station assignment ($X_{t,s}$) provided by SALOME at hand, it is possible to emulate that solution to a fixed task-worker assignment ($W_{t,w}$), treating such variables as parameters.

The residual problem, then, can be reduced to worker-station assignment ($Y_{w,s}$) and task scheduling (ST_t) problems. As precedence relations constraints between stations are automatically respected from the SALBP solution, this model only has to group workers whose assignments do not violate such constraints within each station, and therefore minimizing the number of stations for a given task-worker assignment.

For example, by considering once again the illustrative instance from Table 18, the initial SALBP solution in Figure 20(a), and an upper limit on the number of workers per station set to $NW = 3$, this intermediary step of minimizing stations given task-worker assignments can aggregate multiple workers in the same station while still respecting precedence relations constraints. An intermediary multi-manned feasible solution for the illustrative instance is presented in Figure 21. The initial decomposition procedure is not able to deliver a solution with a reduced production rate (i.e. $CT = 11$, with times on x-axis), and idle times remain at 5 time units along the line, or 9.09% of the line's available task performing time. On the other hand, by allowing co-occurring operations within the third station by three different workers (represented by bars), the intermediary MALBP feasible solution was capable of reducing the line length from 5 to 3 stations (on the y-axis). Notice that all task-worker assignments persist from the input SALBP solution (Figure 20(a)) and no constraint imposed by precedence relations is violated.

As demonstrated, this approach does not ensure optimality for any type of MALBPs. Nonetheless, it has been successfully applied to the MALBP-1 instances by Lopes *et al.* (2019),

Figure 21 – Gantt diagram representation of an intermediary MALBP-2 feasible solution for the illustrative instance by applying the initial decomposition procedure.



Source: Michels *et al.* (2020).

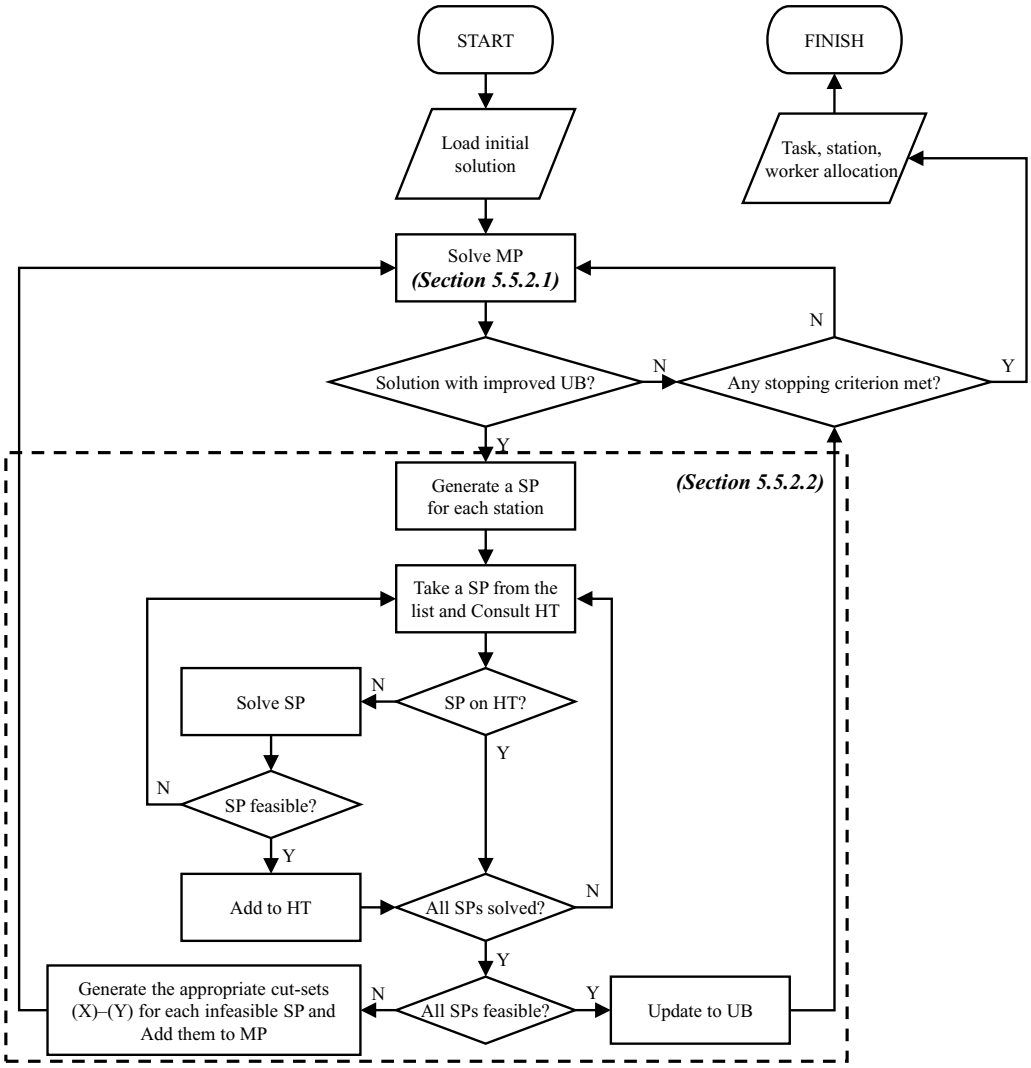
as it yields good solutions in quite reduced computational times. As evidenced in Section 5.6.2, using such task-station-worker assignment as an initial solution for the complete Benders' decomposition approach (Section 5.5.2) can drastically improve faster convergence for the algorithm (*Enhancement 1*).

5.5.2 Benders' decomposition algorithm

For the MALBP-2, the MP tackles task and worker assignments to stations, whereas the SP takes care of task scheduling problems, orderly assigning tasks to workers. Naturally, MP and SP cannot be formulated by simply decoupling parts of the PF: Sections 5.5.2.1 and 5.5.2.2 explain how each of these models is generated. Figure 22 is a flowchart that illustrates the path taken by the proposed BDA, when looking for an optimal solution, in which HT is a hash-table to be consulted for repetitive solutions. When the algorithm loads the initial solution, it consequently establishes an upper bound for the problem and gets a proxy on the direction that the search process should occur: for the illustrative instance, this initial solution is represented by Figure 21, which the proposed algorithm promptly leads to an optimal MALBP-2 solution previously presented in Figure 20(b).

In order to deal with the problem stated in Section 5.4 by the PF (Expressions 60–77), one should notice that the SP is a mixed-integer problem in this application, thus a feasibility-seeking variant (CODATO; FISCHETTI, 2006) should be used. Once again, the SP must be employed as a feasibility check on the algorithm, as analogously proposed by recent MALBP-1 works (MICHELS *et al.*, 2019; NADERI *et al.*, 2019). Given its resources to each multi-manned station, the SPs should, particularly for the proposed BDA, assign and schedule tasks to workers

Figure 22 – General flowchart scheme of the MALBP-2 proposed solution method.



Source: Michels *et al.* (2020).

for each individual station, in an effort to complete all the assigned tasks within the cycle time limit previously defined by the current MP iteration.

The second enhancement is detailed in Section 5.5.2.1: feasibility cuts are implemented and applied to the MP before execution starts, constraining the possibilities of task assignments (*Enhancement 2*). These are based on precedence graph analyzes (KLEIN, 2000). Besides, each infeasible SP returns combinatorial inequalities (cutting planes) to be appended as lazy constraints into the MP, originating the third enhancement procedure to accelerate convergence (CODATO; FISCHETTI, 2006): the use Combinatorial Benders' Cuts (*Enhancement 3*). This is done by extending the previous concept to use such cuts. The MP is distilled from the monolithic model, i.e. the original and complete combinatorial problem, unconstrained from scheduling restriction, since it is initially taken apart from the SP. As all decision variables to calculate

the objective value (CT and Z_s) are found in any given MP iteration, solving it may generate feasible integer solutions. These candidate solutions are then sent to the SP to be validated by it. If all SPs detect the tentative solution to be feasible, the current solution is established as the new incumbent. Otherwise, a set of the aforementioned CBC is returned from each infeasible SP. Finally, the BDA iterates this procedure until (i) an optimal solution is found and proven or (ii) the time limit is reached.

The last improvement prevents redundant feasibility seeking tests to occur (*Enhancement 4*). In order to do so, a hash-table (MAURER; LEWIS, 1975), hereafter referred to as HT, is exploited to store SP instance data that had led to feasible solutions. If a set of task assignments for a given SP is solved and deemed to be feasible, such set of tasks is included into a HT with tested feasible problems, along with the number of workers and cycle time used to perform them. This procedure allows the SP model to skip repeated scheduling problems, since the HT can be quickly consulted by the algorithm's sub-routine beforehand. For similar reasons, every time an infeasibility is detected by an SP, such circumstance is modeled as a new set of constraints and appended to the MP, so those task and worker assignments will not be repeated. These combinatorial Benders' cuts are further explained in Section 5.5.2.2.

5.5.2.1 Master problem

Expressions 60, 61, 64, and 71–77 are kept in the MP, since they are part of the balancing core of the complete problem. Likewise, additional constraints are developed to adapt and strengthen the MP. Before establishing them, however, Table 21 newly introduces the terminology needed to describe the remaining of the MP model.

Table 21 – Definition of new MALBP-2 parameters, sets, and variables for the MP.

Parameter	Description
δ_{t_i,t_j}	Critical path duration between tasks t_i and t_j; $t_i \prec t_j$
σ_{t_i,t_j}	Sum of task durations between tasks t_i and t_j; $t_i \prec t_j$
Set	**Description**
P^*	Set of complete extended precedence relations P: (t_i,t_j)
Su_t	Set of all direct and indirect successor of task t
Pr_t	Set of all direct and indirect predecessor of task t
Inc	Set of incompatible task pairs: (t_i,t_j)
Variable	**Description**
$A_{w,s}$	Auxiliary integer variable: detects the available time for each worker w in station s

Source: Michels *et al.* (2020).

Independently of the number of workers assigned to a station, there are tasks that

cannot be performed together in the same station due to limitations imposed by cycle time and precedence relation restrictions. In order to represent those, a set of incompatible task pairs (Inc) is developed. Algorithm 2 shows the steps on how this incompatibility is determined to find such task pairs. Firstly, the direct precedence relations set P must be extended to a complete (direct and indirect) precedence set P^* in order to do so. That is easily made possible by regarding all direct and indirect precedence relations in the precedence graph. With P^* at hand, the next step consists in constructing sets for each task containing all their successors and predecessor: Su_t represents direct and indirect successors of task t and Pr_t constitutes direct and indirect predecessors of task t. Moreover, the critical path duration and the sum of task durations between tasks t_i and t_j are represented by the two extra parameters δ_{t_i,t_j} and σ_{t_i,t_j}, respectively. A topological order is applied in order to recursively evaluate and properly compute the values of these parameters.

Algorithm 2 – Pseudo-code for the MALBP-2 incompatibility set generation.

1: **function** Incompatibility($P^*, Su_t, Pr_t, CT_{SALBP}, D_t$)
2: $\delta_{t_i,t_j} \leftarrow 0, \sigma_{t_i,t_j} \leftarrow 0, Inc \leftarrow \{\}$
3: **forall** $(t_i,t_j) \in P^*$ **do**
4: $\delta_{t_i,t_j} = \max [0;\ \delta_{t_i,t_k} + D_{t_k} \mid t_k \in Su_{t_i} \cap Pr_{t_j}]$
5: $\sigma_{t_i,t_j} = \sum_{t_k \in Su_{t_i} \cap Pr_{t_j}} D_{t_k}$
6: **if** $D_{t_i} + D_{t_j} + \delta_{t_i,t_j} > CT_{SALBP} \vee D_{t_i} + D_{t_j} + \left\lceil \frac{\sigma_{t_i,t_j}}{NW} \right\rceil > CT_{SALBP}$ **then**
7: $Inc \leftarrow Inc \bigcup \{(t_i,t_j)\}$
8: **end if**
9: **end forall**
10: **return** Incompatibility($\delta_{t_i,t_j}, \sigma_{t_i,t_j}, Inc$)
11: **end function**

Source: Michels *et al.* (2020).

That stated, P^*, Su_t, Pr_t, CT_{SALBP}, and D_t can be the input for Algorithm 2, whereas δ_{t_i,t_j}, σ_{t_i,t_j}, and Inc are its output. Given a task pair $(t_i,t_j) \in P^*$ (line 3), the code runs through the complete precedence graph and establishes the critical path (δ_{t_i,t_j}, line 4) and capacity bounds (σ_{t_i,t_j}, line 5) between any two tasks that have direct or indirect precedent relations, recursively attributing correct values to the parameters. In other words, Algorithm 2 assigns to δ_{t_i,t_j} the highest value for the sum of task durations that have to be executed between tasks t_i and t_j and to σ_{t_i,t_j} the sum of task durations taken from all tasks that are, at the same time, successors and predecessor of tasks t_i and t_j, respectively. This is a logical concept inherited from Klein (2000) and likewise adopted in previous MALBP-1 works Becker and Scholl (2009), Michels *et al.* (2019), which takes into account similarities between the SP and project scheduling problems. Lastly, by taking the task pair (t_1,t_8) of the precedence graph from Table 18 for instance, these

parameters would be $\delta_{t_1,t_8} = 7$ and $\sigma_{t_1,t_8} = 14$.

If either condition $D_{t_i} + D_{t_j} + \delta_{t_i,t_j} > CT_{SALBP}$ or $D_{t_i} + D_{t_j} + \left\lceil \frac{\sigma_{t_i,t_j}}{NW} \right\rceil > CT_{SALBP}$ is verified (line 6), then the condition to flag an incompatibility is satisfied, and the task pair (t_i, t_j) is included into the incompatibility set Inc (line 7). This process goes on until the whole complete precedence graph is analyzed.

Once in possession of Inc, Constraints 78 can be appended to the MP, which should restrict specific task pairs to be assigned to the same station. Generating a set of weaker but still valid inequalities is possible by using information extracted from parameter σ_{t_i,t_j}: a very small positive number (ε) is introduced to prevent dividing by zero, Constraints 79 define what is the minimum number of workers that must be employed in any given station to perform both t_i and t_j of a task pair (t_i, t_j).

$$X_{t_i,s} + X_{t_j,s} \leq 1 \qquad \forall\, s \in S,\ (t_i,t_j) \in Inc \tag{78}$$

$$\sum_{w \in W} Y_{w,s} \geq \left\lceil \frac{\sigma_{t_i,t_j}}{CT_{SALBP} - D_{t_i} - D_{t_j} + \varepsilon} \right\rceil - NW \cdot (2 - X_{t_i,s} - X_{t_j,s}) \qquad \forall\, s \in S,\ (t_i,t_j) \in P^* \tag{79}$$

Finally, precedence relations constraints (Constraints 80) are added to the MP and the available time to execute tasks in each station (Constraints 81) is reckoned. Notice that the available time is dependent on the cycle time and the number of workers assigned for a given station, leading to a non-linear constraint. To solve that issue, an integer auxiliary variable ($A_{w,s}$) must be introduced to evaluate how much available time each station s has at each worker position w. Constraints 82 and 83 are respectively activated as trivial lower and upper bounds for $A_{w,s}$ and limit its value to zero unless some worker w is assigned to their position. Constraints 84 and 85, on the other hand, attribute the correct amount of cycle time to $A_{w,s}$ whenever worker w is employed in a given station. Ultimately, Constraints 81 can be replaced by its linearized version: Constraints 86.

$$\sum_{s \in S} s \cdot X_{t_i,s} \leq \sum_{s \in S} s \cdot X_{t_j,s} \qquad \forall\, (t_i,t_j) \in P \tag{80}$$

$$\sum_{t \in T} X_{t,s} \cdot D_t \leq CT \cdot \sum_{w \in W} Y_{w,s} \qquad \forall\, s \in S \tag{81}$$

$$A_{w,s} \geq \left\lceil \frac{\sum_{t \in T} D_t}{Nmax} \right\rceil \cdot Y_{w,s} \qquad \forall\, (w,s) \in WS \tag{82}$$

$$A_{w,s} \leq CT_{SALBP} \cdot Y_{w,s} \qquad \forall\,(w,s)\ \in\ WS \tag{83}$$

$$A_{w,s} \geq CT - CT_{SALBP} \cdot (1 - Y_{w,s}) \qquad \forall\,(w,s)\ \in\ WS \tag{84}$$

$$A_{w,s} \leq CT - \left\lceil \frac{\sum_{t \in T} D_t}{Nmax} \right\rceil \cdot (1 - Y_{w,s}) \qquad \forall\,(w,s)\ \in\ WS \tag{85}$$

$$\sum_{t \in T} X_{t,s} \cdot D_t \leq \sum_{w \in W} A_{w,s} \qquad \forall\, s\ \in\ S \tag{86}$$

For the proposed BDA, seeking an optimal solution is the emphasis of this reformulated MP. In order to concentrate the efforts in that direction, each feasible solution $(\bar{X}, \bar{Y}) = \{(\bar{X}_{1,1}, ..., \bar{X}_{t,s}), (\bar{Y}_{1,1}, ..., \bar{Y}_{w,s})\}$ with an improved objective value (CT^*, S^*) attained by the MP is passed on to the SP model, so it can carry on scheduling feasibility checks in each station $s \in S^*$ (Figure 22). Therefore, the PF is decomposed into an MP that decides cycle time, task-station allocation, worker-station assignment, and station opening variables: CT, $X_{t,s}$, $Y_{w,s}$, and Z_s, respectively. Meanwhile, the SPs look for feasible task-worker assignments ($W_{t,w}$), by accounting for task starting times and ordering (ST_t and F_{t_i,t_j}) for each station.

5.5.2.2 Slave problem

The path taken by the BDA to solve each SP is presented in Figure 22. Algorithm 3 details each step of that process. It gets as input an improved solution from the MP, with information regarding the task and worker assignments to stations ($\bar{X}$,$\bar{Y}$), as well as the new objective value to be tested, represented by a cycle time (CT^*) and the total number of stations (S^*) used for such solution. The current incumbent (UB), lower bound (LB), and the parameter CT_{SALBP} are also informed.

As to generate an SP for each station $s \in S^*$ (line 3), the SP model maintains Expressions 62, 65, and 70 as in Section 5.4, but simplifies Constraints 67 and adds Constraints 88 and 89 as valid inequalities for idle time symmetry breaks.

For that, they are separately applied to each SP (line 4), where task and worker subsets (T_s and W_s) are extracted from the MP solution $(\bar{X}, \bar{Y})$, as declared by on lines 5 and 6. Regarding each station $s\ \in\ S^*$ as a detached resource-constrained scheduling problem,

Algorithm 3 – Pseudo-code for the MALBP-2 SP generated cuts.

1: **function** SP_Cuts($(\bar{X},\bar{Y}), CT^*, S^*, UB, LB, CT_{SALBP}$)
2: $SP_s \leftarrow infeasible$, $Status \leftarrow 0$, $Counter \leftarrow 0$, $HT \leftarrow \{\}$
3: Generate an SP for each station: SP_s
4: **forall** $s \in S^*$ **do**
5: $T_s = \{t \in T \mid \bar{X}_{t,s} = 1\}$
6: $W_s = \{w \in W \mid \bar{Y}_{w,s} = 1\}$
7: **if** $\{(T_s, W_s, Obj^*)\} \subset HT \vee |W_s| = 1$ **then**
8: $SP_s \leftarrow feasible$
9: $Counter \leftarrow Counter++$
10: **else**
11: Solve SP_s
12: **if** $SP_s = feasible$ **then**
13: $HT \leftarrow HT \bigcup \{(T_s, W_s, CT^*)\}$
14: $Counter \leftarrow Counter++$
15: **else** Generate $Cuts$ for infeasible SP_s
16: **if** $|W_s| = NW \wedge CT^* = CT_{SALBP}$ **then**
17: Add Cut 90
18: **else**
19: **if** $|W_s| < NW \wedge CT^* = CT_{SALBP}$ **then**
20: Add Cut 91
21: **else**
22: **if** $|W_s| = NW \wedge CT^* < CT_{SALBP}$ **then**
23: Add Cut 92
24: **else** ($|W_s| < NW \wedge CT^* < CT_{SALBP}$)
25: Add Cut 93
26: **end if**
27: **end if**
28: **end if**
29: **end if**
30: **end if**
31: **end forall**
32: **if** $Counter = S^*$ **then**
33: Update incumbent solution with a complete $TWS_{allocation}$: $UB \leftarrow (CT^*, S^*)$
34: **if** $UB = LB \vee CPU \geq Time.Limit$ **then**
35: $Status \leftarrow 1$
36: **end if**
37: **end if**
38: **return** SP_Cuts($Cuts, TWS_{allocation}$)
39: **end function**

Source: Michels *et al.* (2020).

Constraints 87, 88, and 89 can replace the previous PF's monolithic ones.

$$F_{t_i,t_j} + F_{t_j,t_i} \geq W_{t_i,w} + W_{t_j,w} - 1 \qquad \forall\, t_i, t_j \in T_s, w \in W_s \tag{87}$$

$$I_w = CT - \sum_{t \in T_s} D_t \cdot W_{t,w} \qquad \forall\, w \in W_s \tag{88}$$

$$I_w \geq I_{w-1} \qquad \forall\, w \in W_s \mid w > 1 \tag{89}$$

There are two situations that an SP is automatically deemed feasible (line 7 to 9): (i) the solution under analysis has already been evaluated before and is contained in HT or (ii) there is

only one worker employed in the station, which means there is no scheduling problem in the first place. If none of these conditions is observed, then the new SP is solved (line 11). Whenever feasibility is detected to this new problem, its information concerning tasks, workers, and cycle time used in the instance is stored into HT (line 13) in order to avoid wasting time solving the same instance in the future by simply accessing HT. On the other hand, when the SP is detected to be infeasible, a set of combinatorial Benders' cuts (CBC) can be applied to the MP as lazy constraints, i.e. the initial optimization problem receives new restrictions while executing in order to cut off infeasible combinations of task and worker assignments to all stations (lines 15 to 29).

Nonetheless, deciding which set of CBCs to add depends on the number of workers ($|W_s|$) and the trial cycle time (CT^*) value used by SPs that were proven to be infeasible. The strongest set of CBCs is appended when the trial solution employs the maximum allowed workers (NW) on a maximum allowed cycle time (CT_{SALBP}) for a specific station (line 16). On line 17, Constraints 90 state that, if any given task assignment set is tested to be infeasible, then it cannot be entirely executed in the same station s, thus at least one of the tasks must be performed elsewhere. Alternatively, the SP may be infeasible, but it might not be using the maximum number of workers for that set of tasks (line 19), the trial cycle time might be lower than the SALBP's imposed one (line 22), or both (line 24).

In such cases, the set of CBCs described by Constraints 91, 92, and 93 are applied to the MP; they respectively state that a tested task assignment set cannot be totally performed in the same station unless an additional worker (represented by the $|W_s| + 1$ index) is assigned there (line 20), the cycle time is increased (line 23), or both (line 25). This wide variety of cuts could not be found in any previous work.

$$\sum_{t \in T_s} X_{t,s} \leq \sum_{t \in T} \bar{X}_{t,s} - 1 \qquad \forall\, s \ \in \ S \tag{90}$$

$$\sum_{t \in T_s} X_{t,s} \leq \sum_{t \in T} \bar{X}_{t,s} - 1 + Y_{|W_s|+1,s} \qquad \forall\, s \ \in \ S \tag{91}$$

$$\sum_{t \in T_s} X_{t,s} \leq \sum_{t \in T} \bar{X}_{t,s} - 1 + (CT - CT^*) \qquad \forall\, s \ \in \ S \tag{92}$$

$$\sum_{t \in T_s} X_{t,s} \leq \sum_{t \in T} \bar{X}_{t,s} - 1 + Y_{|W_s|+1,s} + (CT - CT^*) \qquad \forall\, s \ \in \ S \tag{93}$$

The SP evaluates a trial solution as wholly feasible only if it is feasible for all stations $s \in S^*$ (line 32), then the proposed method judges this solution sent from the MP as incumbent and the instance's UB is updated with the new, improved cycle time, number of stations, and a complete set of task-worker-station assignments (line 33). This process is repeated iteratively and the MP keeps searching for better solutions, either with a revised UB or newly added lazy constraints, until any stopping criterion is met, namely (i) an optimal solution is found and proven or (ii) the CPU time limit is reached (line 34).

In combination with the initial solution generation technique (Section 5.5.1), the BDA herein presented (Section 5.5.2) is what constitutes the whole proposed solution procedure, which is hereafter referred to as PM (Proposed Method).

5.6 COMPUTATIONAL STUDY

This section carries out a computational study based on the same benchmark dataset and industrial case used by Roshani and Giglio (2017). Nevertheless, the current dataset extends the previous one (with 72 instances) by considering the possibility of employing 2, 4, or 6 workers in the same station, totaling 108 instances. A well-known literature benchmark is contained in the tested instances. It is also accessible for download at <www.assembly-line-balancing.de>. They comprise information about task durations, precedence graphs, and cycle time values. The list of instances is summarized in Table 22; in it, instances are divided into three categories: small, medium, and large, according to the number of tasks (NT) parameter. Additionally, there are 4 case study instances regarding the problem studied by Dimitriadis (2006). These instances are run with different values of total number of workers in the line ($Nmax$) and maximum number of workers allowed in each station (NW). As explained in Section 5.4.3, upper bounds for the cycle time (CT) and number of stations (NS) have been obtained with a preliminary process, retrieving information from SALBP-2 counterpart solutions of each instance.

The computational tests are split in three parts. Firstly, Section 5.6.1 reports results obtained by the monolithic model (PF) presented in Section 5.4 when it is applied to small-size instances. These results are compared to those described in Roshani and Giglio (2017); this last mathematical model is henceforth referred to as RF (Roshani's Formulation). In addition, the same fraction of the benchmark dataset is also solved by the PM displayed in Section 5.5, whilst its performance is compared to the best outcome obtained by either Direct or Indirect Simulated Annealing (DSA/ISA) heuristics developed by Roshani and Giglio (2017) in terms of solution

Table 22 – Summary of MALBP-2 dataset instances.

Size (Total of instances)	Problem	NT	$Nmax$	NW
Small (32)	Mitchell	21	3; 4; 5; 7; 8; 9	2
	Heskiaoff	28	4; 6; 8; 9; 10	2; 4
	Sawyer	30	4; 6; 8; 9; 10; 12; 13; 14	2; 4
Medium (34)	Kilbridge	45	4; 6; 8; 10; 11	2; 4
	Tonge	70	12; 14; 16; 18; 19; 20; 22; 23	2; 4; 6
Large (42)	Arcus1	83	11; 12; 14; 15; 18; 19; 21	2; 4; 6
	Arcus2	111	12; 16; 18; 23; 24; 26; 27	2; 4; 6
Case study (4)	Dimitriadis	64	8; 10	2; 4

Source: Michels *et al.* (2020).

quality reported by them; the latter method is henceforth referred to as RM (Roshani's Method).

Secondly, as both methods demonstrated dominance over their respective mathematical formulations, monolithic models (PF and RF) were discarded in the remainder testing process, as they are not expected to keep up with specific methods in terms of solution quality and computational processing time. Therefore, for this medium and large-size computational study conducted afterwards, only PM and RM were considered and applied to MALBP-2 instances. Moreover, in order to test the synergies between the initial decomposition (Dec) and the Benders' decomposition algorithm (BDA) that compose the PM, Dec and BDA were also applied to medium and large-size instances separately. Section 5.6.2 presents and discusses such results and comparisons.

These comparisons are conducted as suitably as possible: as Roshani and Giglio (2017) fixed the number of stations and minimized cycle time as the primary objective, their formulation tends to fit as many workers as it can to the limited number of stations, which leads to solutions with lower efficiencies than its less flexible version. As argued in Section 5.4.3, the assumption of limited number of stations (line area) is not practical and ignore the fact that workers and stations have completely different costs, with those of the former being much higher. Wages are paid to the workforce at least in a monthly basis, so labor costs directly influence the final cost of the product and impose a limit on the resources one can use. Assembly line plants for the automotive industry can regularly surpass 300,000m^2 in area, with extreme examples such as the Hyundai Ulsan Factory and the Volkswagen Wolfsburg Plant with 5,050,000m^2 and 6,500,000m^2 in area, respectively. Considering the dimensions of a regular car and the additional space to perform tasks by multiple workers, a station is a much more plentiful resource than skilled labor. That is precisely why objective functions in the type-1 problem make sense, with workers representing higher costs than stations with a difference of one or two orders of magnitude (FATTAHI; ROSHANI, 2011). Acknowledging that, PF and PM may have outperformed RF and RM in some

instances due to such space limitation imposed by Roshani and Giglio (2017). Nevertheless, specific instances for direct comparison are identified and examined in both Sections 5.6.1 and 5.6.2.

Lastly, Section 5.6.3 evaluates the performance of both methods (PM and RM) when applied to a real-life automotive assembly plant case study. Their efficiency improvements are compared based on instances that contain a dataset originally published by Dimitriadis (2006). Since Roshani and Giglio (2017) also solved a case by fixing the upper limit on the number of workers in the line to be $Nmax = 8$, the comparison here is decidedly straightforward. Furthermore, the other case ($Nmax = 10$) yielded a solution with workers hired for all possible positions in the station, which equally allows a direct comparison with the PM.

In all Sections 5.6.1, 5.6.2, and 5.6.3, each instance result is reported in a line of Tables 23 to 26. They are addressed by their precedence graph structure, $Nmax$, and NW values. For the results, columns headlines with CT, St, Gap, CPU, DCPU, SCPU, CBC, and HT respectively indicate: best found cycle time (CT), best found total number of stations (St) for that cycle time, integrality gap (Gap), computational processing time in total (CPU), for the initial decomposition (DCPU), for the slave problems (SCPU), total number of applied combinatorial Benders' cuts (CBC), and how many times the hash-table has been accessed (HT) for each instance. For small-size instances, Gap is not reported because PM has found the optimal solution for all of them. Some results are left unfilled (–) due to the fact that no integer solutions could be found by the BDA within the time limit for some instances. As this is an extended dataset, the same happens for some instances that were not solved in Roshani and Giglio (2017). For the case study, line efficiency (LE) is also computed.

For every instance, Gurobi 8.1 (Gurobi Optimization, 2019) with optimality focus for the MP and feasibility focus for SPs was employed as universal solver. Four threads of a 64-bit Intel™ i7-3770 CPU (3.4 GHz) with 16GB of RAM were used to run the algorithm. The PM's BDA and its interaction with Dec were coded in Microsoft Visual Basic 2019 programming language. Roshani and Giglio (2017) solved their instances on a 64-bit Intel™ i3-330M CPU (2.13 GHz) with 4GB of RAM. Therefore, a factor of 0.37 (obtained by Passmark Performance Test 9.0) will be applied to their CPU time. Nonetheless, notice that computational processing time results are just mentioned, while relevant comparisons are strictly focused on solution quality.

5.6.1 Small-size instances

Table 23 is a summary of the comparison between monolithic models (PF and RF) and solution methods (PM and RM). In this section, they were applied to the small-size instances with a time limit set to 3600 seconds. Whenever an instance had been solved by Roshani and Giglio (2017) as well, results obtained by PF are compared to those found in the referred paper. Complementary, PM results are exhibited alongside with the best result reported by either DSA or ISA in their respective paper (RM).

The PF clearly outperforms RF in this subset containing 32 instances: the former obtained 16 optimal solutions (50% of cases), whilst the latter reached optimality in only 6 out of the 16 instances tested by Roshani and Giglio (2017), being 4 of them in the smallest problem. In other words, PF has improved 8 previously known integer solutions obtained by a mathematical model and has proven the optimality of 4 of them. This may be attributed to the fact that PF and RF took different modeling decisions: as reported in Section 5.4, the proposed formulation prioritizes cycle time minimization taking into account information acquired from SALBP bounds.

A particular attention is given to the Sawyer family of instances. For $Nmax = 4$, solutions reported for both formulations and methods comprise 2 stations, however, PF and PM were able to prove an optimal cycle time result of 81 time units, while RF and RM were outperformed by yielding solutions with 82 and 83 cycle time units, respectively. By just considering the solution methods, PM has also outperformed RM in instances with $Nmax = 8$ and $Nmax = 12$: PM has proven cycle times of 41 and 28 time units, whereas RM had reached solutions with 42 and 29 cycle time units for the same number of workers and stations along the line.

The remaining information in Table 23 compares PM and RM results. Similar computational processing times were reported for both methods. Nevertheless, the PM presented in Section 5.5 has improved and proven 12 previously known integer solutions (boldfaced values in Table 23). Therefore, it can be stated that the PM has reached better results than RM's heuristics. Moreover, it has guaranteed solutions to be optimal for all small-size instances in a very low CPU time, achieving 100% of optimality proofs in this subset of instances.

Table 23 – Results for MALBP-2 small-size instances: comparison between monolithic formulations (PF and RF) and solution methods (PM and RM).

Problem	$Nmax$	NW	PF			RF			PM							RM		
			CT	St	CPU	CT	St	CPU[1]	CT	St	CPU	DCPU	SCPU	CBC	HT	CT	St	CPU[1]
Mitchell	3	2	35	3	0.07	39	2	0.97	**35**	3	0.07	0.05	0.00	0	0	39	2	17.06
	4	2	27	3	0.19	27	3	1.60	27	3	0.16	0.02	0.01	0	0	27	3	14.33
	5	2	21	5	0.13	–	–	–	**21**	5	0.19	0.01	0.00	0	0	27	3	2.64
	7	2	16	6	0.87	16	6	5.01	16	6	0.26	0.01	0.00	0	0	16	6	12.32
	8	2	14	7	0.05	14	7	6.35	14	7	0.21	0.02	0.00	0	1	14	7	2.58
	9	2	13	8	0.10	13	8	9.32	13	8	0.26	0.01	0.00	0	0	13	8	0.29
Heskiaoff	4	2	256	3	22.01	274	2	1h	**256**	3	32.23	0.01	31.97	1326	1	274	2	6.31
		4	256	3	25.26	–	–	–	256	3	36.81	0.01	36.56	1401	1	–	–	–
	6	2	171	4	1h	183	3	1h	**171**	4	0.05	0.02	0.00	0	2	183	3	7.50
		4	171	4	1h	–	–	–	171	4	0.04	0.01	0.01	0	2	–	–	–
	8	2	129	5	1h	139	4	1h	**129**	5	0.31	0.02	0.03	20	3	139	4	6.87
		4	129	5	1h	–	–	–	129	5	0.34	0.02	0.04	35	3	–	–	–
	9	2	116	7	100.71	124	5	1402.28	**116**	7	1.53	0.02	0.30	749	14	124	5	41.66
		4	116	7	140.03	–	–	–	116	7	2.73	0.02	0.86	1148	13	–	–	–
	10	2	108	6	13.85	108	6	63.26	108	6	0.27	0.02	0.18	318	59	108	6	2.91
		4	108	6	20.99	–	–	–	108	6	0.93	0.02	0.46	804	84	–	–	–
Sawyer	4	2	81	2	446.71	82	2	1h	**81**	2	0.27	0.02	0.02	4	0	83	2	5.93
		4	81	2	108.45	–	–	–	81	2	0.42	0.02	0.01	0	0	–	–	–
	6	2	55	4	1h	56	3	1h	**55**	4	0.22	0.01	0.00	0	2	57	3	7.08
		4	55	4	1h	–	–	–	55	4	0.26	0.01	0.00	0	2	–	–	–
	8	2	41	4	1h	41	4	123.49	**41**	4	0.13	0.01	0.02	7	3	42	4	7.36
		4	41	4	1h	–	–	–	41	4	0.29	0.01	0.02	7	2	–	–	–
	9	2	37	6	1h	39	5	1h	**37**	6	1.82	0.01	0.01	0	5	40	5	7.16
		4	37	6	1h	–	–	–	37	6	2.23	0.01	0.09	24	1	–	–	–
	10	2	34	6	1h	–	–	–	34	6	6.34	0.02	0.05	80	12	34	6	32.15
		4	34	6	1h	–	–	–	34	6	8.08	0.02	0.04	80	10	–	–	–
	12	2	28	7	1h	28	7	1h	**28**	7	0.39	0.02	0.01	0	4	29	7	19.81
		4	28	7	1h	–	–	–	28	7	0.22	0.02	0.01	0	4	–	–	–
	13	2	26	8	1h	–	–	–	**26**	8	2.84	0.02	0.02	10	8	28	7	7.49
		4	26	8	1h	–	–	–	26	8	1.83	0.02	0.01	0	7	–	–	–
	14	2	25	8	16.50	25	8	1h	25	8	1.35	0.02	0.02	36	6	25	8	4.42
		4	25	8	32.05	–	–	–	25	8	1.62	0.02	0.03	27	5	–	–	–

[1] As reported in Roshani and Giglio (2017) multiplied by a factor of 0.37.

Source: Michels *et al.* (2020).

5.6.2 Medium and large-size instances

By rapidly finding and proving optimal solutions in all small instances (Section 5.6.1, Table 23), the PM has been validated as an efficient and reliable method. Besides, its superiority over the monolithic model was evidenced. Hence, only specialized methods (i.e. PM and RM) are compared in this section regarding solution quality. The comparison only takes into account instances that were previously reported in Roshani and Giglio (2017), focusing on computationally solving medium and large-size instances. Furthermore, in order to observe possible synergy effects caused by integrating the initial decomposition (Dec) and the Benders' decomposition algorithm (BDA) that compose the PM, they were also applied to medium and large-size instances separately.

Table 24 reports results for 34 medium-sized instances from Table 22. Out of that enlarged dataset, Roshani and Giglio (2017) has only solved and reported solutions for 13 instances. In terms of solution quality, the PM has outperformed RM in 12 instances (boldfaced in Table 24), whilst tying in the last one. Out of the totality of 34 medium-sized instances, the PM has solved 15 cases to optimality (i.e. 44.12% of this subset), with a minor integer gap for the remaining 19 solutions (1.81% in the worst case and 0.77% on average).

The importance of the possibility to allow 4 workers per station was evidenced for both Kilbridge and Tonge families of instances. By defining $NW = 4$ as a parameter, it has been possible to further reduce the required number of stations along the line in all 13 instances with $NW = 2$. Since this flexibility was not considered by Roshani and Giglio (2017), it can be seen as a novel contribution to the MALBP-2 literature. Furthermore, by closely analyzing the Tonge family last instance ($Nmax = 23$), one can realize that, even for the same number of workers and stations, PM demonstrated to be superior to RM in terms of cycle time, the former resulting in a solution with 156 time units, while the latter only achieved a solution with 177 time units. It is, in relative terms, a 11.86% improvement generated by the PM.

Synergies between Dec and BDA could be verified in many instances throughout the medium dataset. Although PM and BDA have been able to optimally solve the same instances, CPU time reduction was observed in the most challenging ones. In addition, PM found better integer solutions than BDA in 2 instances, and also yielded good integer solutions to 8 instances in which BDA was not capable of attaining any initial solution. It means an improved result in 29.41% of the medium-size instances when compared to the direct BDA approach. Such

improvement can be attributed to Dec feeding the PM with a feasible solution at the beginning of its execution. Nonetheless, also notice that, for those same 8 instances that BDA could not find any feasible solution, PM did not simply maintained the initial solution provided by Dec, but in fact improved all of them in terms of the number of used stations.

The large-size subset from Table 22 contains the last 42 instances to be tested. The comparison between Dec, BDA, PM, and RM when they are applied to such instances is presented in Table 25. The boldfaced values represent 14 results in which the PM has outperformed RM in terms of solution quality for the instances solved and reported in Roshani and Giglio (2017). In total, PM has optimally solved 12 large-sized instances (i.e. 28.57% of them for this subset), whilst reaching integer solutions with gaps inferior to 0.14% (or 0.03% on average) in the remaining 30 ones. Synergies between Dec and BDA were strongly verified this time: (i) out of the 9 instances that both PM and BDA solved to optimality, a reduced CPU time could once again be observed for PM; (ii) out of the remaining 33 instances, BDA was unable to find a feasible solution, whereas PM yielded good integer solutions for all of them, which translates into obtaining improved results for 78.57% of the large-size instances when compared to the direct BDA approach; and (iii) the PM has improved the initial solution found by Dec in 23 instances.

These results convey the efficiency of the PM when applied to the studied problem: similar methods used in isolation in the past for MALBP-1 (LOPES *et al.*, 2020; NADERI *et al.*, 2019; MICHELS *et al.*, 2019) were not sufficient to generate solutions as good as the ones produced by the PM, and the known MALBP-2 solutions previously found by RM (ROSHANI; GIGLIO, 2017) were greatly improved.

Another feature that could be examined with these tests herein conducted was to evidence the possibility to improve cycle time beyond the SALBP-2 optimal solution limit. For MALBP-1, Michels *et al.* (2019) has stated that the results obtained for the problem's type-1 variant were an indicative that it would be more profitable to accept SALBP-1 optimal solutions as the number of workers, while trying to minimize the line length as much as possible, since no improvement could be verified in the total number of workers throughout the entire dataset. Indeed, such methodology generates great results, as corroborated by Lopes *et al.* (2019), Lopes *et al.* (2020). Nevertheless, cycle time improvements could have been confirmed in 12 instances (3 medium and 9 large) for the MALBP-2 dataset, supporting with empirical evidence that the possibility depicted in Figure 20, Section 5.3 exists for benchmark instances as well.

Table 24 – Results for MALBP-2 medium-size instances: comparison between solution methods (PM and RM).

Problem	$Nmax$	NW	Dec			BDA			PM							RM		
			CT	St	DCPU	CT	St	CPU	CT	St	Gap	CPU	SCPU	CBC	HT	CT	St	CPU[1]
Kilbridge	4	2	138	3	0.09	138	3	1h	**138**	3	0.02%	1h	1h	1128	1	146	2	9.60
		4	138	3	0.06	138	2	1.95	138	2	0.00%	2.46	2.03	0	1	–	–	–
	6	2	92	5	0.04	92	4	95.74	**92**	4	0.00%	96.66	91.90	5680	837	98	3	8.64
		4	92	5	0.06	92	3	0.11	92	3	0.00%	1.05	0.65	5	1	–	–	–
	8	2	69	5	0.04	69	5	2.35	**69**	5	0.00%	2.68	2.30	185	22	75	4	7.87
		4	69	5	0.04	69	4	0.14	69	4	0.00%	0.55	0.27	10	2	–	–	–
	10	2	56	7	0.03	56	6	0.54	**56**	6	0.00%	0.96	0.26	35	3	60	5	7.47
		4	56	7	0.03	56	5	0.15	56	5	0.00%	0.29	0.04	7	3	–	–	–
	11	2	55	7	0.06	55	6	1h	55	6	1.81%	1h	6.15	6216	992	55	6	7.35
		4	55	7	0.04	55	5	1h	55	5	1.81%	1h	427.77	251370	55578	–	–	–
Tonge	12	2	294	9	0.11	294	7	1h	**294**	7	0.34%	1h	76.59	2106	180	311	6	46.14
		4	294	9	0.11	294	6	1h	294	6	0.34%	1h	68.61	1008	40	–	–	–
		6	294	9	0.11	294	6	1h	294	6	0.34%	1h	609.12	5319	913	–	–	–
	14	2	251	10	0.12	251	8	514.44	**251**	8	0.00%	282.67	3.34	290	43	280	7	33.78
		4	251	10	0.11	251	7	965.28	251	7	0.00%	901.05	225.78	2430	483	–	–	–
		6	251	10	0.11	251	7	773.48	251	7	0.00%	409.16	25.01	690	36	–	–	–
	16	2	221	14	0.16	220	10	733.06	**220**	10	0.00%	360.79	42.96	10346	1231	240	8	33.17
		4	221	14	0.17	220	9	1928.28	220	9	0.00%	1022.80	66.70	5166	487	–	–	–
		6	221	14	0.16	221	7	1h	220	9	0.01%	1h	51.02	4998	298	–	–	–
	18	2	196	14	0.09	–	–	1h	**196**	13	0.51%	1h	6.45	4381	257	225	9	37.41
		4	196	13	0.11	–	–	1h	196	12	0.52%	1h	19.64	5447	276	–	–	–
		6	196	13	0.10	–	–	1h	196	11	0.51%	1h	14.85	4108	183	–	–	–
	19	2	186	14	0.16	–	–	1h	**186**	12	0.53%	1h	5.42	4200	161	206	10	147.49
		4	186	14	0.16	–	–	1h	186	11	0.54%	1h	70.82	7518	607	–	–	–
		6	186	14	0.16	–	–	1h	186	11	0.54%	1h	45.60	7140	385	–	–	–
	20	2	177	14	0.11	177	12	235.86	**177**	12	0.00%	228.95	14.29	9264	1152	197	10	70.78
		4	177	12	0.11	177	11	2479.63	177	11	0.00%	1581.09	63.97	12924	1704	–	–	–
		6	177	12	0.11	177	11	3567.05	177	11	0.00%	2692.13	78.77	14664	1854	–	–	–
	22	2	162	16	0.24	–	–	1h	**162**	15	1.24%	1h	11.45	13536	1206	178	12	117.11
		4	162	16	0.24	–	–	1h	162	11	1.22%	1h	40.16	11632	686	–	–	–
		6	162	16	0.24	162	11	1h	162	11	1.23%	1h	20.44	9392	518	–	–	–
	23	2	156	14	0.09	156	12	1h	**156**	12	0.64%	1h	10.62	23016	5701	177	12	22.75
		4	156	14	0.09	156	10	1h	156	10	1.24%	1h	59.62	25844	2724	–	–	–
		6	156	14	0.09	156	12	1h	156	10	1.25%	1h	22.22	18130	1479	–	–	–

[1]As reported in Roshani and Giglio (2017) multiplied by a factor of 0.37.

Source: Michels *et al.* (2020).

Table 25 – Results for MALBP-2 large-size instances: comparison between solution methods (PM and RM).

Problem	$Nmax$	NW	Dec			BDA			PM							RM		
			CT	St	DCPU	CT	St	CPU	CT	St	Gap	CPU	SCPU	CBC	HT	CT	St	CPU[1]
Arcus1	11	2	7084	9	0.09	–	–	1h	**7084**	9	0.00%	897.11	58.97	171	7	7909	6	311.65
		4	7084	9	0.09	–	–	1h	7084	9	0.00%	710.37	59.31	189	9	–	–	–
		6	7084	9	0.09	–	–	1h	7084	9	0.00%	1911.08	62.45	153	9	–	–	–
	12	2	6412	11	0.10	–	–	1h	**6412**	11	0.02%	1h	57.36	693	17	7688	6	83.51
		4	6412	11	0.10	–	–	1h	6412	10	0.03%	1h	87.91	220	5	–	–	–
		6	6412	11	0.10	–	–	1h	6412	10	0.03%	1h	58.17	253	9	–	–	–
	14	2	5441	12	0.08	5441	10	136.83	**5441**	10	0.00%	124.42	1.92	180	5	6412	8	449.18
		4	5441	11	0.09	5441	10	193.60	5441	10	0.00%	181.12	121.97	1012	18	–	–	–
		6	5441	11	0.08	5441	10	260.13	5441	10	0.00%	164.33	11.72	0	4	–	–	–
	15	2	5104	11	0.20	5104	10	347.04	**5104**	10	0.00%	221.40	48.24	1309	85	6194	8	81.15
		4	5104	11	0.29	5104	9	1658.65	5104	9	0.00%	996.66	115	913	48	–	–	–
		6	5104	11	0.19	5104	9	1413.57	5104	9	0.00%	1270.48	350.75	1287	69	–	–	–
	18	2	4317	17	1.46	–	–	1h	**4317**	17	0.14%	1h	0.42	102	4	4811	10	85.61
		4	4317	17	1.45	–	–	1h	4317	17	0.14%	1h	0.16	136	3	–	–	–
		6	4317	17	1.45	–	–	1h	4317	16	0.14%	1h	0.16	85	5	–	–	–
	19	2	4068	16	0.32	–	–	1h	**4068**	16	0.03%	1h	1.73	608	5	4494	12	61.93
		4	4068	16	0.32	–	–	1h	4068	16	0.03%	1h	11.28	912	14	–	–	–
		6	4068	16	0.31	–	–	1h	4068	16	0.03%	1h	8.88	496	23	–	–	–
	21	2	3691	18	3.20	3691	14	1144.26	**3691**	14	0.00%	275.22	5.24	3924	175	4320	12	252.04
		4	3691	17	3.20	3691	14	1155.77	3691	14	0.00%	555.81	16.80	2822	58	–	–	–
		6	3691	17	3.20	3691	14	1240.87	3691	14	0.00%	187.77	7.11	527	29	–	–	–
Arcus2	12	2	12534	9	9.25	–	–	1h	**12534**	9	0.01%	1h	3412.53	2835	57	13350	6	158.48
		4	12534	9	9.60	–	–	1h	12534	9	0.01%	1h	3277.14	1800	20	–	–	–
		6	12534	9	9.66	–	–	1h	12534	9	0.01%	1h	3241.73	2106	67	–	–	–
	16	2	9412	12	6.75	–	–	1h	**9412**	12	0.01%	1h	0.01	0	12	10889	8	219.91
		4	9412	12	6.80	–	–	1h	9412	12	0.01%	1h	0.01	0	12	–	–	–
		6	9412	12	6.76	–	–	1h	9412	12	0.01%	1h	34.62	300	16	–	–	–
	18	2	8377	14	1’	–	–	1h	**8377**	14	0.02%	1h	211.07	840	23	9726	10	193.71
		4	8377	13	1’	–	–	1h	8377	11	0.02%	1h	1197.72	11414	48	–	–	–
		6	8377	13	1’	–	–	1h	8377	11	0.02%	1h	321.29	1001	10	–	–	–
	23	2	6561	17	1’	–	–	1h	**6560**	17	0.03%	1h	1815.44	16150	48	7467	13	161.03
		4	6561	17	1’	–	–	1h	6560	17	0.06%	1h	974.15	7871	29	–	–	–
		6	6561	17	1’	–	–	1h	6560	17	0.05%	1h	1428.07	9554	37	–	–	–
	24	2	6313	18	1’	–	–	1h	**6310**	17	0.02%	1h	2635.00	17280	75	7235	13	793.65
		4	6313	18	1’	–	–	1h	6310	17	0.03%	1h	2661.06	15750	121	–	–	–

[1]As reported in Roshani and Giglio (2017) multiplied by a factor of 0.37.

Source: Michels *et al.* (2020).

The full assignment and scheduling results concerning tasks, workers, and stations were tested for all solutions in order to carry on with a feasibility check. The starting time for each task was evaluated for consistency in regard to station and worker assignments, global cycle times, and order imposed by precedence relations. These assignment files are made available in Michels *et al.* (2020)'s supporting information for reproducibility purposes, filling their function of validating the proposed PM's reliability.

Finally, contrary to the BDA applied to MALBP-1 (MICHELS *et al.*, 2019), in most cases the PM has a tendency to spend the majority of its available CPU time running the MP. The latter is more similar to the case of the BDA proposed by Naderi *et al.* (2019), which spends more than 98% of its computational processing time solving the MP.

5.6.3 Real-life assembly plant case study

The PM is lastly applied to the data of an industrial assembly plant originally published by Dimitriadis (2006) and its results are compared to those obtained by the RM: Table 26 summarizes the results of such comparison. The number of tasks to be performed in this case study is 64, with a total duration time of 65693 time units. In this study, Roshani and Giglio (2017) also fixed the total number of workers, lines with 8 and 10 total workers were tested, allowing at most 2 or 4 workers per station in each case.

Table 26 – Results for MALBP-2 real-life case study instances: comparison between solution methods (PM and RM).

Prob	$Nmax$	NW	PM								RM		
			CT	St	LE	CPU[1]	DCPU	SCPU	CBC	HT	CT	St	LE
Dimit	8	2	8212	4	99.9%	7.3	9.7	5.5	0	3	8310	5	98.8%
		4	8212	3	99.9%	1h	10.0	1h	108	2	–	–	–
	10	2	6571	5	99.9%	1h	1'	95.3	1945	155	6650	5	98.6%
		4	6570	3	99.9%	191.5	1'	191.5	6	4	–	–	–

[1]Not reported in Roshani and Giglio (2017).

Source: Michels *et al.* (2020).

According to these results, the PM methodology is once again more effective than RM when it comes to determining lower values of cycle time. In fact, for $NW = 2$, PM reduced the cycle time (improved the line efficiency) for $Nmax = 8$ and $Nmax = 10$ from 8310 (98.8%) to 8212 (99.9%) and from 6650 (98.6%) to 6571 (99.9%), respectively, with the same numbers of workers in both cases and one less stations in the former. Moreover, when $NW = 4$ is allowed, line length is reduced to 3 stations in both situations, with improvements in cycle time (6571 to

6570) and line efficiency (99.9%) for $Nmax = 10$.

5.7 CONCLUSIONS

The type-2 Multi-manned Assembly Line Balancing Problem (MALBP-2) has been addressed in this study. The objective is to minimize the cycle time and the total number of stations as ranked goals. The existing literature on MALBP-2 was very limited, and efficient exact solution methods were only available for the type-1 variant (MALBP-1). This chapter's main contribution was presenting an innovative method to optimality solve larger MALBP-2 instances by decomposing the original problem. It proposed an initial solution procedure and implemented a Benders' decomposition algorithm with the application of multiple combinatorial Benders' cuts during its execution. The results showed that the proposed algorithm is very efficient in comparison to an adapted version of an exact method for the MALBP-1 and to a meta-heuristic for the MALBP-2.

In order to solve the optimization problem, a new Mixed-Integer Linear Programming (MILP) model was developed (Section 5.4.1), along with several symmetry break constraints (i.e. valid inequalities, Section 5.4.2) and strong bounds (Section 5.4.3). The proposed formulation (PF) outperformed previous monolithic mathematical formulations in terms of solution quality and computational processing time (Section 5.6.1). By studying MALBP-1 specific solutions methods, it was possible to infer that the type-2 variant was also hierarchically divisible into a Master Problem (MP) and a group of Slave Problems (SPs). By reformulating the original monolithic model and using problem specific knowledge, a new Benders' Decomposition Algorithm (BDA) with an initial feasible solution procedure is forged (Section 5.5). The solutions obtained by the proposed method (PM) were compared to previously developed methods by applying the PM to an extended benchmark dataset and a real-life assembly plant case study (Section 5.6.1). From the 46 instances that an integer solution was known in the literature, the PM was able to produce improved results in most cases, totaling 38 new integer solutions, of which 29 were proven optimal. In total, 60 optimal solutions were obtained out of a dataset with 108 instances, resulting in 55.56% of optimal solutions in the proposed dataset (Section 5.6.2). Lastly, better integer solutions were found in all case study instances (4), of which 2 were deemed optimal (Section 5.6.3).

Industries manufacturing large-size products often allow multiple tasks to be simultaneously performed by different workers in the same station. Given the resources, the managers aim

to achieve the best possible productivity level. This is a notable realistic feature, which is viewed as a natural extension of the problem's simpler version and is widely employed in the literature. Nonetheless, incorporating more practical extensions is a desirable modification. For instance, equipment selection, worker heterogeneity, product variability (multi and mixed-model lines), and line layouts (U-line, parallel stations) could be valuable features to be added to the PM. Further research should concentrate in doing so, as well as implementing balancing and project scheduling heuristics for the master and slave problems to mitigate computational burden.

6 CONCLUDING REMARKS

This chapter presents final considerations along with the main contributions of the presented PhD thesis. Furthermore, future research directions are provided.

6.1 FINAL CONSIDERATIONS AND CONTRIBUTIONS

This PhD thesis proposed solution methods that combine mathematical programming and decomposition algorithms to solve extensions of the assembly line balancing problem. Several real-world operational aspects of the studied industrial assembly lines have been taken into account, such as robotic assembly lines and multi-operated stations. The obtained solutions are analysed and compared to real-world as-built robotic lines or previous results from benchmark datasets.

Chapter 1 introduced usual assembly lines found in manufacturing industries (e.g. Figure 1 on page 14), focusing on the primary importance of decision aid models for assembly line balancing problems. Several design and operational costs are concerned: facilities, equipment, robots, workforce, productivity levels, among others. Furthermore, the general and specific objectives of this PhD thesis had been described and the author's research output has been listed.

The details and main terminologies of these particular assembly line balancing problems were described in Chapter 2, such as its elements, physical characteristics, operational constraints, and typical simplification hypotheses. Moreover, an extensive literature review based on publications and surveys regarding the Assembly Line Balancing Problem (ALBP) was conducted in Chapter 2 to further define the proposed problems. This chapter specifically covered the definition and uses of flow-shop production layouts, which are employed in automotive assembly lines and gives rise to the ALBP. The simplification hypotheses (SH) presented in Section 2.2 define the most explored problem in the literature, the Simple Assembly Line Balancing Problem (SALBP). Several variations of SALBPs were discussed in Section 2.3. In order to generalise the problem into a General Assembly Line Balancing Problem (GALBP), SH are relaxed and exemplified. A further generalisation was explored in Section 2.4 to define the Assembly Line Design Problem (ALDP). Variations that are applicable in the automotive industry were presented, describing their distinctive characteristics. Finally, in conjunction with introductory sections of Chapters 3, 4, and 5, robotic assembly lines and multi-operated workstations are thoroughly

examined in Sections 2.5 and 2.6 to place the proposed problems into the gap and evidence its contribution.

Contribution 1: The development of a literature review concerning the ALBP and its general variations to evidence gaps for (i) an automotive industry assembly line design model and (ii) exact solution methods for assembly lines with multi-manned stations.

Contributions of Chapter 3 are separately highlighted in Section 6.1.1. Similarly, contributions of Chapters 4 and 5 are independently listed in Section 6.1.2.

6.1.1 Contributions to the robotic assembly line literature

Providing the best solution to real-world problems has been the greatest interest of optimisation practical applications. Nonetheless, a gap between the academy research and industrial applications still exists in the literature, one of them was observed on robotic welding assembly lines, which are frequently found in the automotive industry: defining their production layout design is an important global and strategic decision. In Chapter 3, the work developed in the full-paper Michels *et al.* (2018b) and published in the *Computers & Industrial Engineering* is presented. The Robotic Assembly Line Design (RALD) problem is defined and an MILP formulation is proposed for it, taking into account several practical considerations of an RALD scenario. The proposed model incorporates the linearisation of a cubic constraint and allows to explicitly evaluate costs and benefits associated to parallel stations in an exact manner.

In Section 3.2, the RALD problem is described in detail and an overview of the optimisation elements is done. It states the problem's assumption and distinguishes the proposed problem from previous Robotic Assembly Line Balancing (RALB) problems. Furthermore, it described practical advantages of parallel station and welding tasks characteristics found in automotive industries.

Contribution 2: The proposal and definition of the Robotic Assembly Line Design (RALD) problem, which considers real-world practical extensions.

An MILP formulation to solve the proposed RALD problem was exposed in Section 3.3. The modelling challenges were due to an integer linearisation of a cubic constraint and the possibility to allow parallel stations in order to weight the costs and benefits of this feature in an exact manner. Moreover, extra practical restrictions caused by welding line characteristics are added to the general problem.

Contribution 3: The development of an MILP formulation to solve the proposed problem and the modelling of extra restriction for resistance spot welding tasks.

Furthermore, the model described in Section 3.3 was applied to computational tests and case studies in Section 3.4, in which the results were presented and discussed. Computational case studies were performed in Section 3.4.1, combining large instances of real-world inspired cases adapted from Sikora *et al.* (2017b) and cost ratio principles proposed by Askin and Zhou (1997). The main conclusions drawn from this experiment were: (i) optimal answers tend to have more parallel stations as the dead time increases or the robots are costly compared to the equipment, and (ii) the intense use of track-motion devices when equipment prices are much higher than the robot ones, due to its tendency to be more cost-effective.

Contribution 4: The development of a validation dataset for the parameters' influence analysis and the model's computational results analyses themselves.

Practical case studies based on three vehicle models presented in Sikora *et al.* (2017b) reached optimal solutions (Section 3.4.2). As previously indicated in Section 3.4.2.1, it has been possible only because the third vehicle model line layout could assemble both vehicle models 1 and 2. Furthermore, parallel stations evidenced its essential role when unproductive times are considered, though paralleling was not necessarily cost-effective in every condition (e.g. Figure 13, on page 62).

Contribution 5: The presentation of an industrial case study from a real-world automotive assembly line to apply the RALD model and the possibility to design robotic assembly lines at lower costs in the future.

6.1.2 Contributions to the multi-manned assembly line literature

In Chapters 2, 4, and 5, the existing literature on MALBPs referenced in Sections 2.6, 4.1, and 5.2 indicated a lack of efficient exact solution methods for these problems.

The type-1 Multi-manned Assembly Line Balancing Problem (MALBP-1) with the objective of minimising the number of workers and stations has been addressed in Chapter 4 of this PhD thesis. Preliminary results of this work were shortly presented in Michels *et al.* (2018a), and its current version (MICHELS *et al.*, 2019) is published in the *European Journal of Operational Research*.

In Section 4.3, a new Mixed-Integer Linear Programming (MILP) model was developed along with several valid inequalities that work as symmetry break constraints to solve the optimisation problem. This proposed formulation outperforms previously presented monolithic mathematical formulations (KELLEGÖZ, 2017) in terms of solution quality and computational processing time (Section 4.5.1).

Contribution 6: The development of a new decomposable MILP model with valid inequalities for the MALBP-1.

By analysing the structure of MALBPs, it was possible to infer that the problem was hierarchically divisible into a Master Problem (MP) and a Slave Problem (SP). Hence, a Benders' Decomposition Algorithm (BDA) could be forged (Section 4.4). The proposed BDA was compared to previously developed methods (KELLEGÖZ, 2017) and was shown to produce improved results while maintaining reasonable CPU time (Section 4.5.2).

Contribution 7: The generation of a decomposition strategy for the MALBP-1 to forge a Benders' decomposition algorithm.

Since past mathematical formulations were only able to solve some instances with up to 45 tasks, the main contribution of this work (MICHELS *et al.*, 2019) was the ability to optimally solve MALBPs up to 148 tasks. It has been possible by decomposing the original problem and implementing a Benders' decomposition algorithm employing combinatorial cuts during its execution.

Contribution 8: The development of an exact algorithm to optimally solve large-size instances from the MALBP-1 benchmark.

Correspondingly, the type-2 Multi-manned Assembly Line Balancing Problem (MALBP-2) with the objective of minimising the line's cycle time and number of stations has been approached in Chapter 5 of this PhD thesis. The manuscript containing this variant of the MALBP is presented in a slightly modified version (MICHELS *et al.*, 2020) from the one currently published in the *Operations Research Perspectives*.

In Section 5.4, the Mixed-Integer Linear Programming (MILP) model developed for the MALBP-1 is extended to the MALBP-2 along with appropriate valid inequalities for the problem at hand. This section explores crucial modelling decisions to correctly define the MALBP-2 in the literature.

Contribution 9: The clarification on the definition of what constitutes a MALBP-2.

The proposed formulation for the MALBP-2 also outperforms previously presented monolithic mathematical formulations (ROSHANI; GIGLIO, 2017) in terms of solution quality and computational processing time (Section 5.6.1).

Contribution 10: The extension of the previous MILP model with valid inequalities for the MALBP-2.

Section 5.5 presented the proposed method for the MALBP-2: it integrated an initial solution procedure with an adapted and enhanced BDA from the previous work. When compared to previously developed methods (ROSHANI; GIGLIO, 2017), it has produced improved results for virtually all medium and large-size instances (Section 5.6.2).

Contribution 11: The adaptation, extension, and enhancement of the previously developed Benders' decomposition algorithm for the MALBP-2.

Once again, past mathematical formulations were only able to optimally solve few instances with up to 30 tasks, the main contribution of this work (MICHELS *et al.*, 2020) was to develop a method that is capable to optimally solve MALBP-2 instances up to 83 tasks. It has only been possible by integrating an initial solution procedure with an adapted and enhanced Benders' decomposition algorithm from the previous MALBP-1 work. Innovative Benders' combinatorial cuts were developed and synergies between the initial procedure and the enhanced BDA were verified.

Contribution 12: The development of an exact algorithm to optimally solve large-size and real-world instances from the MALBP-2 benchmark.

6.2 FUTURE RESEARCH

The study published in Michels *et al.* (2018b) exposed how effective the proposed MILP formulation is when it comes to designing a robotic assembly line. It has included practical extensions that were able to shrink the search-space. Therefore, for future research, the proposed model can be widened to incorporate task scheduling for each robot in the station. Moreover, the model may be adapted to represent literature variants, such as different product models characteristics in a mixed-model line and set-up times between them.

Allowing multiple workers to simultaneously perform different tasks at the same station is a natural extension of the simpler version of the assembly line balancing problem. It is also a notable realistic feature widely employed in industries manufacturing large-size products.

Nonetheless, incorporating more practical extensions into the developed BDAs is a desirable modification to extend works developed in Michels *et al.* (2019) and Michels *et al.* (2020). Different line layouts (U-line, parallel stations), product variety (multi and mixed-model lines), and task assignment restriction are desirable extensions to be modified in the BDAs.

Further research should focus on doing so and mitigating computational burden at the same time. Despite the quite significant progress achieved by the proposed models and algorithms, further important improvements can still be devised for new methods. They might include, for instance, balancing and project scheduling heuristics for the master and slave problems, respectively. That stated, as a final research direction of this thesis, the author invites readers to enhance and integrate the previously developed models and algorithms as follows:

- MALBP: integrating equipment selection, paralleling possibilities, and dead time as an extension to the classical problem. This may generate a cost-oriented MALBP, thus several adaptations are also required in the previously proposed BDAs.
- RALD: integrating task scheduling for each robot in the station in order to generalise the problem for all types of tasks. It guarantees that multiple robots working at the same station respect precedence relations within stations, which could translate into practical features being included into the enhanced MALBP model.

In addition to the mentioned items, there is room for improvements to the presented algorithms and MILP models themselves. That could undoubtedly contribute to facilitate applications, improve solution quality, and enhance performance of the proposed methods.

REFERENCES

AKAGI, Fumio; OSAKI, Hirokazu; KIKUCHI, Susumu. A method for assembly line balancing with more than one worker in each station. **International Journal of Production Research**, v. 21, n. 5, p. 755–770, 1983. ISSN 1366588X.

AKPINAR, Sener; ELMI, Atabak; BEKTAş, Tolga. Combinatorial Benders cuts for assembly line balancing problems with setups. **European Journal of Operational Research**, v. 259, n. 2, p. 527–537, 2017. ISSN 03772217.

AMEN, Matthias. Heuristic methods for cost-oriented assembly line balancing: A survey. **International Journal of Production Economics**, v. 68, n. 1, p. 1–14, 2000.

AMEN, Matthias. Cost-oriented assembly line balancing: Model formulations, solution difficulty, upper and lower bounds. **European Journal of Operational Research**, v. 168, p. 747–770, 2006.

ARAÚJO, Felipe F.B.; COSTA, Alysson M; MIRALLES, Cristóbal. Balancing parallel assembly lines with disabled workers. **European Journal of Industrial Engineering**, v. 9, n. 3, p. 344–365, 2015. ISSN 1751-5254.

ASKIN, R G; ZHOU, M. A parallel station heuristic for the mixed-model production line balancing problem. **International Journal of Production Research**, v. 35, n. 11, p. 3095–3106, 1997.

BARD, Jonathan F. Assembly line balancing with parallel workstations and dead time. **International Journal of Production Research**, v. 27, n. 62015, p. 1005–1018, 1989.

BARTHOLDI, J. J. Balancing two-sided assembly lines: A case study. **International Journal of Production Research**, v. 31, n. 10, p. 2447–2461, 1993. ISSN 1366588X.

BATTAÏA, Olga; DOLGUI, Alexandre. A taxonomy of line balancing problems and their solution approaches. **International Journal of Production Economics**, v. 142, n. 2, p. 259–277, apr 2013. ISSN 09255273.

BAUTISTA, Joaquín; PEREIRA, Jordi. A dynamic programming based heuristic for the assembly line balancing problem. **European Journal of Operational Research**, v. 194, n. 3, p. 787–794, 2009. ISSN 03772217.

BAYBARS, Ilker. A survey of exact algorithms for the simple assembly line balancing problem. **Management Science**, v. 32, n. 8, p. 909–932, 1986.

BAYKASOGLU, Adil; TASAN, Seren Ozmehmet; TASAN, Ali Serdar; AKYOL, Sebnem Demirkol. Modeling and solving assembly line design problems by considering human factors with a real-life application. **Human Factors and Ergonomics In Manufacturing**, v. 27, n. 2, p. 96–115, 2017.

BECKER, Christian; SCHOLL, Armin. A survey on problems and methods in generalized assembly line balancing. **European Journal of Operational Research**, v. 168, n. 3, p. 694–715, 2006. ISSN 03772217.

BECKER, Christian; SCHOLL, Armin. Balancing assembly lines with variable parallel workplaces: Problem definition and effective solution procedure. **European Journal of Operational Research**, v. 199, n. 2, p. 359–374, 2009. ISSN 03772217.

BENDERS, J F. Partitioning procedures for solving mixed-variables programming problems. **Numerische Mathematik**, v. 4, n. 1, p. 238–252, 1962. ISSN 0029599X.

BOWMAN, E. H. Assembly-Line Balancing by Linear Programming. **Operations Research**, v. 8, n. 3, p. 385–389, 1960.

BOYSEN, Nils; FLIEDNER, Malte; SCHOLL, Armin. A classification of assembly line balancing problems. **European Journal of Operational Research**, v. 183, n. 2, p. 674–693, dec 2007. ISSN 03772217.

BOYSEN, Nils; FLIEDNER, Malte; SCHOLL, Armin. Assembly line balancing: Which model to use when? **International Journal of Production Economics**, v. 111, n. 2, p. 509–528, 2008. ISSN 09255273.

BUKCHIN, Joseph; RUBINOVITZ, Jacob. A weighted approach for assembly line design with station paralleling and equipment selection. **IIE Transactions**, v. 35, p. 73–85, 2003.

BUKCHIN, Joseph; TZUR, Michal. Design of fexible assembly line to minimize equipment cost. **IIE Transactions**, v. 32, p. 585–598, 2000.

CAKIR, Burcin; ALTIPARMAK, Fulya; DENGIZ, Berna. Multi-objective optimization of a stochastic assembly line balancing: A hybrid simulated annealing algorithm. **Computers & Industrial Engineering**, v. 60, n. 3, p. 376–384, 2011. ISSN 0360-8352.

CEVIKCAN, Emre; DURMUSOGLU, M Bulent; UNAL, Murat E. A team-oriented design methodology for mixed model assembly systems. **Computers & Industrial Engineering**, v. 56, n. 2, p. 576–599, 2009. ISSN 0360-8352.

CHEN, Yin Yann. A hybrid algorithm for allocating tasks, operators, and workstations in multi-manned assembly lines. **Journal of Manufacturing Systems**, v. 42, p. 196–209, 2017. ISSN 02786125.

CODATO, Gianni; FISCHETTI, Matteo. Combinatorial Benders' Cuts for Mixed-Integer Linear Programming. **Operations Research**, v. 54, n. 4, p. 756–766, 2006. ISSN 0030-364X.

CÔTÉ, Jean-françois; DELL'AMICO, Mauro; IORI, Manuel. Combinatorial Benders' Cuts for the Strip Packing Problem. **Operations Research**, v. 62, n. 3, p. 643–661, 2014. ISSN 0030-364X.

DAOUD, S; CHEHADE, H; YALAOUI, F; LIONEL, Amodeo. Solving a robotic assembly line balancing problem using efficient hybrid mehtods. **Journal of Heuristics**, v. 20, p. 235–259, 2014.

DECKRO, Richard F. Balancing Cycle Time and Workstations. **IIE Transactions**, v. 21, n. 2, p. 106–111, 1989.

DIMITRIADIS, Sotirios G. Assembly line balancing and group working: A heuristic procedure for workers' groups operating on the same product and workstation. **Computers & Operations Research**, v. 33, n. 9, p. 2757–2774, 2006. ISSN 03050548.

DOLGUI, Alexandre; GUSCHINSKY, Nikolai; LEVIN, Genrikh. Enhanced mixed integer programming model for a transfer line design problem. **Computers & Industrial Engineering**, v. 62, n. 2, p. 570–578, 2012. ISSN 0360-8352.

EGE, Yunus; AZIZOGLU, Meral; OZDEMIREL, Nur E. Computers & Industrial Engineering Assembly line balancing with station paralleling. **Computers & Industrial Engineering**, v. 57, n. 4, p. 1218–1225, 2009. ISSN 0360-8352.

ESSAFI, Mohamed; DELORME, Xavier; DOLGUI, Alexandre; GUSCHINSKAYA, Olga. A MIP approach for balancing transfer line with complex industrial constraints. **Computers & Industrial Engineering**, v. 58, n. 3, p. 393–400, 2010. ISSN 03608352.

FAKHRI, Ashkan; GHATEE, Mehdi; FRAGKOGIOS, Antonios; SAHARIDIS, Georgios K D. Benders decomposition with integer subproblem. **Expert Systems with Applications**, v. 89, p. 20–30, 2017. ISSN 09574174.

FATTAHI, Parviz; ROSHANI, Abdolreza. A mathematical model and ant colony algorithm for multi-manned assembly line balancing problem. **International Journal of Advanced Manufacturing Technology**, v. 53, p. 363–378, 2011.

GAO, Jie; SUN, Linyan; WANG, Lihua; GEN, Mitsuo. An efficient approach for type II robotic assembly line balancing problems. **Computers & Industrial Engineering**, v. 56, n. 3, p. 1065–1080, 2009. ISSN 0360-8352.

GOKÇEN, Hadi; AGPAK, Kursad; BENZER, Recep. Balancing of parallel assembly lines. **International Journal of Production Economics**, v. 103, p. 600–609, 2006.

GUNEY, Yusuf; AHISKA, S Sebnem. Automation Level Optimization for an Automobile Assembly Line. *In*: **Proceedings of the 2014 Industrial and Systems Engineering Research Conference**. *[S.l.: s.n.]*, 2014. p. 2209–2218.

Gurobi Optimization. **Gurobi Optimizer reference manual**. 2019. 789 p.

HAZIR, Öncü; DOLGUI, Alexandre. Assembly line balancing under uncertainty: Robust optimization models and exact solution method. **Computers & Industrial Engineering**, v. 65, p. 261–267, 2013.

HAZIR, Öncü; DOLGUI, Alexandre. A decomposition based solution algorithm for U-type assembly line balancing with interval data. **Computers & Operations Research**, v. 59, p. 126–131, 2015. ISSN 03050548.

HILIER, Frederick; LIEBERMAN, Gerald. **Introduction to Operational Research**. 10th. ed. Heidelberg: Mc Graw Hill, 2015. 731–784 p. ISSN 0896-3207. ISBN 978-0-07-352345-3.

HOFFMANN, T R. Assembly line balancing with a precedence matrix. **Management Science**, v. 9, n. 4, p. 551–562, 1963.

JACKSON, J R. A computing procedure for a line balancing problem. **Management Science**, v. 2, n. 3, p. 261–271, 1956.

KAZEMI, Abolfazl; SEDIGHI, Abdolhossein. A cost-oriented model for balancing mixed-model assembly lines with multi-manned workstations. **International Journal of Services and Operations Management**, v. 16, n. 3, p. 289, 2013. ISSN 1744-2370.

KELLEGÖZ, Talip. Assembly line balancing problems with multi-manned stations: a new mathematical formulation and Gantt based heuristic method. **Annals of Operations Research**, v. 253, n. 1, p. 377–404, 2017. ISSN 15729338.

KELLEGÖZ, Talip; TOKLU, Bilal. An efficient branch and bound algorithm for assembly line balancing problems with parallel multi-manned workstations. **Computers & Operations Research**, v. 39, n. 12, p. 3344–3360, 2012. ISSN 03050548.

KELLEGÖZ, Talip; TOKLU, Bilal. A priority rule-based constructive heuristic and an improvement method for balancing assembly lines with parallel multi-manned workstations. **International Journal of Production Research**, v. 53, n. 3, p. 736–756, 2015. ISSN 1366588X.

KIM, H; PARK, S. Strong cutting plane algorithm for the robotic assembly line balancing problem. **International Journal of Production Research**, v. 33, n. 8, p. 2311–2323, 1995. ISSN 00207543.

KIM, Yeo Keun; KIM, Jae Yun; KIM, Yeongho. A Coevolutionary Algorithm for Balancing and Sequencing in Mixed Model Assembly Lines. **Applied Intelligence**, v. 13, p. 247–258, 2000.

KIM, Yeo Keun; KIM, Yeongho; KIM, Yong Ju. Two-sided assembly line balancing: A genetic algorithm approach. **Production Planning & Control: The Management of Operations**, v. 11, n. 1, p. 44–53, 2000.

KLEIN, Robert. **Scheduling of Resource-Constrained Projects**. 1st. ed. New York: *[s.n.]*, 2000. 369 p. ISBN 978-0-7923-8637-7.

KOTTAS, John F; LAU, Hon Shiang. Cost-oriented approach to stochastic line balancing. **AIIE Transactions**, v. 5, n. 2, p. 164–171, 1973.

KRAJEWSKI, L J; RITZMAN, L P; MALHOTRA, M K. **Operations Management: Processes and Supply Chains**. 10th. ed. *[S.l.]*: Pearson, 2013. ISBN 9780132807395.

LEVITIN, Gregory; RUBINOVITZ, Jacob; SHNITS, Boris. A genetic algorithm for robotic assembly line balancing. **European Journal of Operational Research**, v. 168, n. 3, p. 811–825, 2006.

LOPES, Thiago Cantos; MICHELS, Adalberto Sato; LÜDERS, Ricardo; MAGATÃO, Leandro. A simheuristic approach for throughput maximization of asynchronous buffered stochastic mixed-model assembly lines. **Computers and Operations Research**, v. 115, p. 104863, 2020. ISSN 03050548.

LOPES, Thiago Cantos; MICHELS, Adalberto Sato; MAGATÃO, Leandro. A note to: A hybrid algorithm for allocating tasks, operators, and workstations in multi-manned assembly lines. **Journal of Manufacturing Systems**, v. 52, p. 205–208, 2019. ISSN 0278-6125.

LOPES, Thiago Cantos; MICHELS, Adalberto Sato; MOLINA, Rafael Gobi; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. An MILP formulation for robotic assembly line design with parallel stations, dead time and equipment selection. *In*: **XLVIII Simpósio Brasileiro de Pesquisa Operacional**. Vitória - ES: *[s.n.]*, 2016. p. 3257–3258.

LOPES, Thiago Cantos; MICHELS, Adalberto Sato; SIKORA, Celso Gustavo Stall; MOLINA, Rafael Gobbi; MAGATÃO, Leandro. Balancing and cyclically sequencing synchronous, asynchronous, and hybrid unpaced assembly lines. **International Journal of Production Economics**, v. 203, p. 216–224, 2018. ISSN 09255273.

LOPES, Thiago Cantos; MICHELS, Adalberto Sato; SIKORA, Celso Gustavo Stall; SILVA, Arinei Carlos Lindbeck; MAGATÃO, Leandro. Modeling the Stochastic Steady-State of Mixed-Model Asynchronous Assembly Lines with Markov Chains. *In*: **XIX Latin-Iberoamerican Conference on Operations Research**. Lima - Peru: CLAIO, 2018. p. 183–190.

LOPES, Thiago Cantos; MICHELS, Adalberto Sato; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. Balancing and cyclical scheduling of asynchronous mixed-model assembly lines with parallel stations. **Journal of Manufacturing Systems**, v. 50, p. 193–200, 2019. ISSN 02786125.

LOPES, Thiago Cantos; PASTRE, Giuliano Vidal; MICHELS, Adalberto Sato; MAGATÃO, Leandro. Flexible Multi-Manned Assembly Line Balancing Problem: Model, Heuristic Procedure, and Lower Bounds for Line Length Minimization. **Omega**, v. 95, p. 102063, 2020.

LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; MAGATÃO, Leandro. A New Model for Simultaneous Balancing and Cyclical Sequencing of Asynchronous Mixed-Model Assembly Lines with Parallel Stations. *In*: **XLIX Simpósio Brasileiro de Pesquisa Operacional**. Blumenau - SC: SBPO, 2017. p. 1–12.

LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; MAGATÃO, Leandro. A Matheuristic for Makespan Minimization of Asynchronous Stochastic Mixed-Model Assembly Lines. *In*: **L Simpósio Brasileiro de Pesquisa Operacional**. Rio de Janeiro - RJ: SBPO, 2018. p. 1–12.

LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; MAGATÃO, Leandro. An iterative decomposition for asynchronous mixed-model assembly lines: combining balancing, sequencing, and buffer allocation. **International Journal of Production Research**, v. 58, n. 2, p. 615–630, 2020. ISSN 1366588X.

LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; MAGATÃO, Leandro. Mixed-model assembly lines balancing with given buffers and product sequence: model, formulation comparisons, and case study. **Annals of Operations Research**, v. 286, p. 475–500, 2020. ISSN 0254-5330.

LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MOLINA, Rafael Gobbi; SCHIBELBAIN, Daniel; RODRIGUES, L. C. A.; MAGATÃO, Leandro. Balancing a robotic spot welding manufacturing line: An industrial case study. **European Journal of Operational Research**, v. 263, n. 3, p. 1033–1048, 2017. ISSN 03772217.

LUSA, Amaia. A survey of the literature on the multiple or parallel assembly line balancing problem. **European Journal of Industrial Engineering**, v. 2, n. 1, p. 50–72, 2008.

MAGNANTI, T L; WONG, R T. Accelerating Benders Decomposition: Algorithmic Enhancement and Model Selection Criteria. **Operations Research**, v. 29, n. 3, p. 464–484, 1981.

MAURER, W D; LEWIS, T G. Hash Table Methods. **ACM Computing Surveys (CSUR)**, v. 7, n. 1, p. 5–19, 1975. ISSN 03600300.

MICHALOS, G.; MAKRIS, S.; PAPAKOSTAS, N.; MOURTZIS, D.; CHRYSSOLOURIS, G. Automotive assembly technologies review: challenges and outlook for a flexible and adaptive approach. **CIRP Journal of Manufacturing Science and Technology**, v. 2, n. 2, p. 81–91, jan 2010. ISSN 17555817.

MICHELS, Adalberto Sato. **Designing Industrial Production Layouts: An Application On Robotic Welding Assembly Lines**. 2017. 73 p. Master Thesis — Universidade Tecnológica Federal do Paraná, Curitiba, Brazil, 2017.

MICHELS, Adalberto Sato; LOPES, Thiago Cantos; MAGATÃO, Leandro. An exact method with decomposition techniques and combinatorial Benders' cuts for the type-2 multi-manned assembly line balancing problem. **Operations Research Perspectives**, p. 100163, 2020. ISSN 2214-7160.

MICHELS, Adalberto Sato; LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. A new mathematical formulation with search-space reduction techniques for the multi-manned assembly line balancing problem. *In*: **29th European Conference on Operational Research**. Valencia: EURO, 2018. p. 135.

MICHELS, Adalberto Sato; LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. The Robotic Assembly Line Design (RALD) problem: Model and case studies with practical extensions. **Computers & Industrial Engineering**, v. 120, p. 320–333, 2018. ISSN 03608352.

MICHELS, Adalberto Sato; LOPES, Thiago Cantos; SIKORA, Celso Gustavo Stall; MAGATÃO, Leandro. A Benders' decomposition algorithm with combinatorial cuts for the multi-manned assembly line balancing problem. **European Journal of Operational Research**, v. 278, n. 3, p. 796–808, 2019. ISSN 0377-2217.

MICHELS, Adalberto Sato; SIKORA, Celso Gustavo Stall; LOPES, Thiago Cantos; MAGATÃO, Leandro. An MILP Formulation for Local Search k-Opt Improvement Procedures. *In*: **L Simpósio Brasileiro de Pesquisa Operacional**. Rio de Janeiro - RJ: SBPO, 2018. p. 1–12.

MILTENBURG, John. Balancing and Scheduling Mixed-Model U-Shaped Production Lines. **The International Journal of Flexible Manufacturing Systems**, v. 14, p. 119–151, 2002.

MILTENBURG, John; WIJNGAARD, J. U-line line balancing problem. **Management Science**, v. 40, n. 10, p. 1378–1388, 1994. ISSN 00251909.

MIRALLES, Cristóbal; GARCÍA-SABATER, José P; ANDRÉS, Carlos; CARDÓS, Manuel. Branch and bound procedures for solving the Assembly Line Worker Assignment and Balancing Problem: Application to Sheltered Work centres for Disabled. **Discrete Applied Mathematics**, v. 156, n. 3, p. 352–367, 2008. ISSN 0166218X.

MOON, Ilkyeong; LOGENDRAN, Rasaratnam; LEE, Jeonghun. Integrated assembly line balancing with resource restrictions. **International Journal of Production Research**, v. 47, n. 19, p. 5525–5541, 2009. ISSN 00207543.

MOREIRA, Mayron César O; MIRALLES, Cristóbal; COSTA, Alysson M. Model and heuristics for the Assembly Line Worker Integration and Balancing Problem. **Computers and Operations Research**, v. 54, p. 64–73, 2015. ISSN 03050548.

NADERI, Bahman; AZAB, Ahmed; BOROOSHAN, Katayoun. A realistic multi-manned five-sided mixed-model assembly line balancing and scheduling problem with moving workers and limited workspace. **International Journal of Production Research**, v. 57, n. 2, p. 643–661, 2019. ISSN 0020-7543.

OESTERLE, Jonathan; AMODEO, Lionel; YALAOUI, Farouk. A comparative study of Multi-Objective Algorithms for the Assembly Line Balancing and Equipment Selection Problem under consideration of Product Design Alternatives. **Journal of Intelligent Manufacturing**, in press, p. 1–26, 2017.

OSMAN, Hany; BAKI, M F. Balancing transfer lines using Benders decomposition and ant colony optimisation techniques. **International Journal of Production Research**, v. 52, n. 5, p. 1334–1350, 2014. ISSN 00207543.

ÖZCAN, Uğur. Balancing stochastic two-sided assembly lines: A chance-constrained, piecewise-linear, mixed integer program and a simulated annealing algorithm. **European Journal of Operational Research**, v. 205, n. 1, p. 81–97, aug 2010. ISSN 03772217.

ÖZCAN, Ugur; TOKLU, Bilal. Balancing of mixed-model two-sided assembly lines. **Computers & Industrial Engineering**, v. 57, p. 217–227, 2009.

ÖZTÜRK, Cemalettin; TUNALI, Semra; HNICH, Brahim; ÖRNEK, Arslan. Cyclic scheduling of flexible mixed model assembly lines with paralel stations. **Journal of Manufacturing Systems**, v. 36, n. 1, p. 147–158, 2015.

PAPE, T. Heuristics and lower bounds for the simple assembly line balancing problem type 1: Overview, computational tests and improvements. **European Journal of Operational Research**, v. 240, p. 32–42, 2015.

PARK, Kyungchul; PARK, Sungsoo; KIM, Wanhee. A heuristic for an assembly line balancing problem with incompatibility, range, and partial precedence constraints. **Computers & Industrial Engineering**, v. 32, n. 2, p. 321–332, apr 1997. ISSN 03608352.

PINEDO, Michael L. **Scheduling: Theory, Algorithms, and Systems**. 5th. ed. *[S.l.]*: Springer, 2016. 670 p. ISBN 978-3-319-26578-0.

RAHMANIANI, Ragheb; CRAINIC, Teodor Gabriel; GENDREAU, Michel; REI, Walter. The Benders decomposition algorithm: A literature review. **European Journal of Operational Research**, v. 259, n. 3, p. 801–817, 2017. ISSN 03772217.

REKIEK, Brahim; DOLGUI, Alexandre; DELCHAMBRE, Alain; BRATCU, Antoneta. State of art of optimization methods for assembly line design. **Annual Reviews in Control**, v. 26, p. 163–174, 2002.

ROSHANI, Abdolreza; GIGLIO, Davide. Simulated annealing algorithms for the multi-manned assembly line balancing problem: minimising cycle time. **International Journal of Production Research**, v. 55, n. 10, p. 2731–2751, 2017. ISSN 1366588X.

ROSHANI, Abdolreza; NEZAMI, Farnaz Ghazi. Mixed-model multi-manned assembly line balancing problem: A mathematical model and a simulated annealing approach. **Assembly Automation**, v. 37, n. 1, p. 34–50, 2017. ISSN 01445154.

ROSHANI, Abdolreza; ROSHANI, Arezoo; ROSHANI, Abdolhassan; SALEHI, Mohsen; ESFANDYARI, Azadeh. A simulated annealing algorithm for multi-manned assembly line balancing problem. **Journal of Manufacturing Systems**, v. 32, n. 1, p. 238–247, 2013. ISSN 02786125.

RUBINOVITZ, J; BUKCHIN, J. Design and balancing of robotic assembly lines. *In*: **Proceedings of the fourth world conference on robotics research**. Pittsburgh, PA: *[s.n.]*, 1991.

RUBINOVITZ, Jacob; BUKCHIN, Joseph. RALB - A Heuristic Algorithm for Design and Balancing of Robotic Assembly Lines. **Annals of the CIRP**, v. 42, n. 1, p. 497–500, 1993.

SABUNCUOGLU, Ihsan; EREL, Erdal; ALP, Arda. Ant colony optimization for the single model U-type assembly line balancing problem. **International Journal of Production Economics**, v. 120, n. 2, p. 287–300, 2009. ISSN 0925-5273.

SAHIN, Murat; KELLEGÖZ, Talip. A new mixed-integer linear programming formulation and particle swarm optimization based hybrid heuristic for the problem of resource investment and balancing of the assembly line with multi-manned workstations. **Computers & Industrial Engineering**, v. 133, n. March, p. 107–120, 2019. ISSN 03608352.

SAHIN, Murat; KELLEGÖZ, Talip. Balancing Multi-Manned Assembly Lines With Walking Workers: Problem Definition, Mathematical Formulation, and an Electromagnetic Field Optimisation Algorithm. **International Journal of Production Research**, v. 7543, 2019. ISSN 1366588X.

SALVESON, M E. The assembly line balancing problem. **The Journal of Industrial Engineering**, v. 6, p. 18–25, 1955.

SAWIK, Tadeusz. Batch versus cyclic scheduling of flexible flow shops by mixed-integer programming. **International Journal of Production Research**, v. 50, n. 18, p. 5017–5034, 2012.

SCHOLL, Armin. **Balancing and sequencing assembly lines**. 2nd. ed. Heidelberg: ed. Physica, 1999.

SCHOLL, Armin; BECKER, Christian. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. **European Journal of Operational Research**, v. 168, n. 3, p. 666–693, feb 2006.

SCHOLL, Armin; BOYSEN, N; FLIEDNER, M. The sequence-dependent assembly line balancing problem. **Operations Research Spectrum**, v. 30, p. 579–609, 2008.

SCHOLL, Armin; BOYSEN, Nils; FLIEDNER, Malte. The assembly line balancing and scheduling problem with sequence-dependent setup times: problem extension, model formulation and efficient heuristics. **OR Spectrum**, n. 35, p. 291–320, 2013.

SCHOLL, Armin; FLIEDNER, Malte; BOYSEN, Nils. ABSALOM: Balancing assembly lines with assignment restrictions. **European Journal of Operational Research**, v. 200, n. 3, p. 688–701, feb 2010. ISSN 0377-2217.

SCHOLL, Armin; KLEIN, Robert. SALOME: A Bidirectional Branch-and-Bound Procedure for Assembly Line Balancing. **INFORMS Journal on Computing**, v. 9, n. 4, p. 319–334, 1997.

SEWELL, E C; JACOBSON, S H. A Branch, Bound, and Remember Algorithm for the Simple Assembly Line Balancing Problem. **INFORMS Journal on Computing**, v. 24, n. 3, p. 433–442, 2012.

SIKORA, Celso Gustavo Stall; LOPES, Thiago Cantos; LOPES, Heitor Silvério; MAGATÃO, Leandro. Genetic algorithm for type-2 assembly line balancing. *In*: **2015 Latin-America Congress on Computational Intelligence**. Curitiba - PR: LA-CCI, 2015. ISBN 9781467384186.

SIKORA, Celso Gustavo Stall; LOPES, Thiago Cantos; MAGATÃO, Leandro. Traveling worker assembly line (re)balancing problem: model, reduction techniques, and real case studies. **European Journal of Operational Research**, v. 259, p. 949–971, 2017.

SIKORA, Celso Gustavo Stall; LOPES, Thiago Cantos; SCHIBELBAIN, Daniel; MAGATÃO, Leandro. Integer based formulation for the simple assembly line balancing problem with multiple identical tasks. **Computers & Industrial Engineering**, v. 104, p. 134–144, 2017.

SIKORA, Celso Gustavo Stall; MICHELS, Adalberto Sato; LOPES, Thiago Cantos; MAGATÃO, Leandro. Combining k-Opt Improvement Procedure and Tabu-Search to Solve the Symmetric TSP via MILP Formulation. *In*: **XLIX Simpósio Brasileiro de Pesquisa Operacional**. Blumenau - SC: SBPO, 2017. p. 3533.

STERNATZ, J. Enhanced multi-Hoffmann heuristic for efficiently solving real-world assembly line balancing problems in automotive industry. **European Journal of Operational Research**, v. 235, p. 740–754, 2014.

SUWANNARONGSRI, Supaporn; PUANGDOWNREONG, Deacha. Optimal assembly line balancing using tabu search with partial random permutation technique. **International Journal of Management Science and Engineering Management**, v. 3, n. 1, p. 3–18, 2008.

THOMOPOULOS, Nick T. Line Balancing-Sequencing for Mixed-Model Assembly. **Management Science**, v. 14, n. 2, p. 59–75, 1967.

TUNCEL, Gonca; TOPALOGLU, Seyda. Assembly line balancing with positional constraints, task assignment restrictions and station paralleling: A case in an electronics company. **Computers & Industrial Engineering**, v. 64, n. 2, p. 602–609, 2013. ISSN 0360-8352.

WALSH, Toby. General Symmetry Breaking Constraints. **Principles and Practice of Constraint Programming - CP**, v. 4204, p. 650–664, 2006.

WHITE, W W. Comments on a Paper by Bowman. **Operations Research**, v. 9, p. 274–276, 1961.

YADAV, Ashish; KUMAR, Shashank; AGRAWAL, Sunil. Reconfiguration of assembly line balancing–an automobile case study solved by the exact solution procedure. **Benchmarking**, 2020. ISSN 14635771.

YAZGAN, Harun Resit; BEYPINAR, Ismail; BORAN, Semra; OCAK, Ceren. A new algorithm and multi-response Taguchi method to solve line balancing problem in an automotive industry. **International Journal of Advanced Manufacturing Technology**, v. 57, p. 379–392, 2011.

YILMAZ, Hamid; YILMAZ, Mustafa. Multi-manned assembly line balancing problem with balanced load density. **Assembly Automation**, v. 35, n. 1, p. 137–142, 2015. ISSN 01445154.

YILMAZ, Hamid; YILMAZ, Mustafa. A multi-manned assembly line balancing problem with classified teams: A new approach. **Assembly Automation**, v. 36, n. 1, p. 51–59, 2016. ISSN 01445154.

YILMAZ, Hamid; YILMAZ, Mustafa. Note to: a mathematical model and ant colony algorithm for multi-manned assembly line balancing problem. **International Journal of Advanced Manufacturing Technology**, v. 89, n. 5-8, p. 1935–1939, 2016. ISSN 14333015.

YILMAZ, Hamid; YILMAZ, Mustafa. A mathematical model and tabu search algorithm for multi-manned assembly line balancing problems with assignment restrictions. **Engineering Optimization**, v. 52, n. 5, p. 856–874, 2020. ISSN 10290273.

YOOSEFELAHI, Ali; AMINNAYERI, Majid; MOSADEGH, Hadi; ARDAKANI, Hamed Davari. Type II robotic assembly line balancing problem: An evolution strategies algorithm for a multi-objective model. **Journal of Manufacturing Systems**, v. 31, n. 2, p. 139–151, 2012. ISSN 0278-6125.