Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia COPPE UFRJ

# MATHEMATICAL MODELING FOR THE JOB SHOP SCHEDULING PROBLEM WITH REALISTIC CONSTRAINTS

Arthur Santâna da Silva

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientadores: Daniel Ratton Figueiredo
Laura Silvia Bahiense da Silva Leite

Rio de Janeiro
Dezembro de 2023

# MATHEMATICAL MODELING FOR THE JOB SHOP SCHEDULING PROBLEM WITH REALISTIC CONSTRAINTS

Arthur Santâna da Silva

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientadores: Daniel Ratton Figueiredo
Laura Silvia Bahiense da Silva Leite

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2023

Santâna da Silva, Arthur

MATHEMATICAL MODELING FOR THE JOB SHOP SCHEDULING PROBLEM WITH REALISTIC CONSTRAINTS/Arthur Santâna da Silva. – Rio de Janeiro: UFRJ/COPPE, 2023.

XI, 58 p.: il.; 29,7cm.

Orientadores: Daniel Ratton Figueiredo

Laura Silvia Bahiense da Silva Leite

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2023.

Referências Bibliográficas: p. 54 – 58.

1. Job Shop Problem. 2. Production Scheduling. 3. Mixed-Integer Programming. I. Ratton Figueiredo, Daniel *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

*A minha esposa Jéssica, a quem*
*devo mais do que posso*
*mensurar.*

# Agradecimentos

Gostaria de expressar minha mais profunda gratidão a todas as pessoas que tornaram este trabalho possível e que, de alguma forma, contribuíram para o meu crescimento pessoal, profissional e acadêmico.

Primeiramente, agradeço à minha amada esposa, cujo amor, compreensão, apoio, encorajamento e companheirismo foram inestimáveis. Sua empatia, paciência e compreensão durante os momentos mais desafiadores me dão força para que eu consiga continuar prosseguindo, mesmo quando nem tudo está a favor. Muito obrigado meu amor, por me ajudar a ter uma visão mais honesta e clara sobre muitas coisas na vida, por entender os sacrifícios que eu escolhi fazer, para que eu conseguisse dar cada pequeno passo até chegar aqui. A única certeza que tenho na vida é te ter sempre ao meu lado.

Aos meus pais Marcos e Eliana e ao meu irmão Rodrigo, que sempre acreditaram em mim, me apoiaram em todas as minhas aspirações, e continuam me impulsionando e me auxiliando em tudo o que podem. Aos meus amigos que me ajudam a espairecer quando preciso, mas que também me apoiam, em especial, minha cunhada Joyce, que não mede esforços pra oferecer ajuda, e que tem um empenho excepcional em tudo que faz.

Aos meus orientadores Laura e Daniel, cuja sabedoria, conhecimento e orientação foram fundamentais para a realização deste trabalho. Sua dedicação e compromisso com a excelência acadêmica são verdadeiramente inspiradores. Para além disso, são pessoas por quem tenho profunda admiração e me apoiaram para que eu pudesse prosseguir nos momentos turbulentos.

Aos amigos e ex-companheiros de trabalho Andréa, Manoel e Sandro, que sempre me deram todo apoio e condições para que, não só esse trabalho fosse realizado, mas que também me incentivaram a crescer academica, profissional e pessoalmente.

Por último, mas não menos importante, a Deus, por me abençoar com pessoas tão únicas, especiais e maravilhosas ao meu redor, cujo alcance vai além dos nomes aqui escritos, mas que estão gravados na minha pessoa de alguma forma. Os atos mais honestos e singelos que temos uns com os outros ecoam de uma forma inexplicável ao longo da vida, e de alguma forma, carrego comigo essas incríveis lembranças.

A todos, meu profundo agradecimento.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

MODELAGEM MATEMÁTICA PARA O PROBLEMA DE JOB SHOP COM RESTRIÇÕES REALISTAS

Arthur Santâna da Silva

Dezembro/2023

Orientadores: Daniel Ratton Figueiredo
Laura Silvia Bahiense da Silva Leite

Programa: Engenharia de Sistemas e Computação

O problema do job shop flexível determina a programação da fabricação, considerando diferentes tarefas e máquinas. Devido à generalidade e complexidade de diferentes linhas e processos de produção, os modelos para esse problema costumam ser relativamente específicos e restritos. Este trabalho considera o problema de job shop flexível com restrições mais realistas, como horários de início mínimos, diferentes funções objetivo e o uso de Grafos Acíclicos Direcionados (GADs) para representar restrições de precedência entre operações de um mesmo job. Quatro modelos de Programação Inteira Mista (PIM) são propostos para capturar tais cenários, e são resolvidos pelo método de branch-and-bound clássico. No primeiro conjunto de experimentos, um software de simulação a eventos discretos, que lida com vários tipos de problemas de programação de produção e possui um núcleo heurístico, é usado para comparar a qualidade das soluções obtidas. No segundo conjunto de experimentos, o mesmo software é utilizado para gerar soluções viáveis a serem utilizadas pelo solucionador matemático. São avaliadas instâncias de referência, sintéticas e outras baseadas em casos do mundo real. O primeiro conjunto de experimentos numéricos indica que as soluções dos modelos de PIM propostos superam as soluções heurísticas em todos os casos testados. O segundo conjunto de experimentos numéricos indica que a resolução dos modelos pode ser afetada através de melhorias nas soluções dual e primal encontradas ao longo da resolução dos modelos.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

MATHEMATICAL MODELING FOR THE JOB SHOP SCHEDULING PROBLEM WITH REALISTIC CONSTRAINTS

Arthur Santâna da Silva

December/2023

Advisors: Daniel Ratton Figueiredo
Laura Silvia Bahiense da Silva Leite

Department: Systems Engineering and Computer Science

The flexible job shop problem determines manufacturing scheduling taking into account different tasks and machines. Due to the generality and complexity of different production lines and processes, models for this problem are often relatively specific and restricted to a production line. This paper considers the flexible job shop problem with more realistic constraints such as minimum start times, different objective functions and the use of a Directed Acyclic Graph (DAG) to represent precedence constraints between operations in the same job. Four Mixed Integer Programming (MIP) models are proposed to capture these scenarios and benchmark instances are solved using the classical branch-and-bound method. A discrete event simulation software which handles various types of production scheduling problems and has a heuristic core to generate viable solutions is used to compare the quality of the solutions obtained via MIP. In the second scenario, the viable solutions generated by the simulation software are used as initial solutions by the mathematical solver. Reference, synthetic and other instances based on real-world cases are evaluated. The first set of numerical experiments indicate that the solutions obtained from the proposed MIP models outperform the heuristic solutions in all the cases evaluated. The second set of numerical experiments indicate that the MIP models are affected through improvements in the dual and primal solutions found throughout the execution of the branch-and-bound method.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The agile development of industrial production technologies combined with changes in the characteristics of the markets for manufactured products are striking characteristics of modern times, highlighting the constant need for upgrading by manufacturing companies and processes [1, 2]. One of the activities most impacted by these changes is production planning and control. In this sense, especially in recent decades, the search for efficient and effective means of planning and control has been pursued.

The industry originally emerged as a paradigm of producing very standardized products, i.e. without much variety, on a large scale, to be manufactured by dedicate and scarce resources [3, 4]. The purpose of this structure was to reduce unit costs by increasing production efficiency. Since a wider range of items must be produced by the same set of flexible resources [5], the current scenario faces significantly more production scheduling complexity [6, 7]. On the other hand, meeting new demands for customization and product variety requires the flexibility and agility of both resources and production scheduling to keep up with the constantly fluctuating production scenario. Such flexibility contradicts the assumption of mass production stability to pursue the highest possible economic efficiency. So, how do such seemingly opposing goals coexist [8]?

All of these factors amalgamate to make short-term production planning and scheduling critical in manufacturing operations. Ensuring that production goals are met within a relatively short deadline entails coordination and optimization of production processes. A production schedule determines what jobs are to be executed in what machines at what time while satisfying a diverse set of constraints. Planning requires tracking inventory levels and modifying production schedules in response to shifts in demand, supply, or other production limitations such as machine failure [9]. Thus, planning and scheduling are often jointly addressed in manufacturing operations.

To be successful, companies must properly plan and manage their scarce resources during manufacturing. A good plan and schedule can guarantee the desired amount and quality is produced at the appropriate time, while also reducing waste, downtime, and other costs. Thus, constructing effective plans and schedules is fundamental for modern manufacturing, and this gives rise to the Job Shop Scheduling Problem (JSSP). Indeed, there is a large body of literature in tackling these problems using a broad variety of problem formulation [10–20].

Many real-world manufacturing problems are also incredibly complex and very specific in terms of constraints and actions. Nevertheless, recent academic works have addressed planning and scheduling problems that increasingly resemble real-world scenarios [21]. A typical strategy is to optimize the production plan using mathematical programming techniques such as Mixed-Integer Programming (MIP). The capacity of examining alternative production schedules and selecting the most efficient schedule is a key benefit of employing MIP in JSSP.

Manufacturers can use mathematical models to plan production efficiently (or optimally) and allocate resources by considering production constraints like finite resources or machine capacity in complex production environments [22]. Furthermore, such models can be updated during production to reflect changes in the manufacturing environment, allowing manufacturers to adjust their schedules in real-time to account for new constraints or shifting customer demand. Thus, MIP is an indispensable tool for modern manufacturing processes.

## 1.1 Motivation

Scheduling a production line becomes incredibly challenging due to countless interconnected factors that create a complex and dynamic environment. A historical and comparative analysis of two paradigms of industrial management has been discussed: mass production and flexible/agile production DUGUAY *et al.* [3]. Mass production which emerged in the nineteenth century and dominated the twentieth century, is characterized by a mechanistic and rigid approach to reducing costs, increasing volume, and controlling change. Mass production relies on innovation by experts and managers, division of labor, and adversarial relations with suppliers. In contrast, flexible/agile production which emerged in the 1980s as a response to the increasing complexity and dynamism of the environment, is characterized by an organic and adaptive approach to improving quality, time, and cost simultaneously. Flexible production relies on continuous improvement by empowered workers, teamwork, and partnership with suppliers. These paradigms are illustrated with examples from the automobile industry and other sectors DUGUAY *et al.* [3]. They conclude that flexible production is the dominant paradigm in industrial management for the twenty-first century.

All of these changes resulted in an increase in complexity in the manufacturing process, which had an impact on the complexity of the mathematical models that were designed to take these characteristics into account. In the JSSP each product may have different processing requirements, manufacturing times, and dependencies on specific machines or resources. As a result, creating an efficient schedule that can accommodate this diverse mix of products while optimizing the use of resources becomes a formidable task that must be automated.

Figure 1.1 depicts an example of non-optimized production scheduling. Figure 1.2 depicts the same scenario with some scheduling improvements, such as shorter operation completion times. If we had all of the necessary scenario parameters, we could easily perform the calculations manually and determine the optimum value.

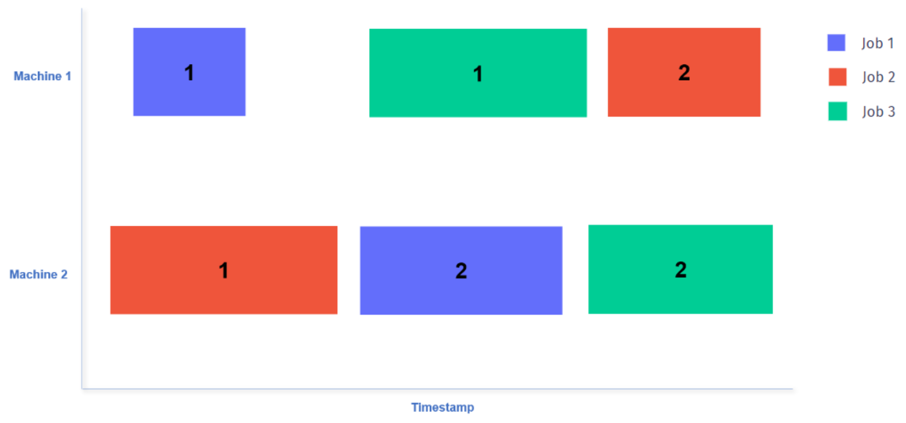

Figure 1.1: An example of production scheduling.

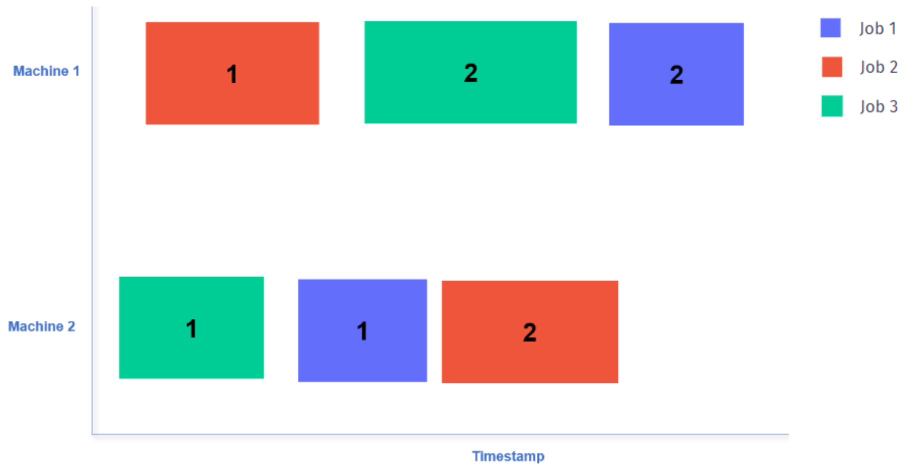

Figure 1.2: An example of production scheduling with minor improvement.

However, this problem can easily reach a point where such manual computations are no longer feasible in enough time in order to obtain a schedule with satisfactory objective values, as shown in Figure 1.3.

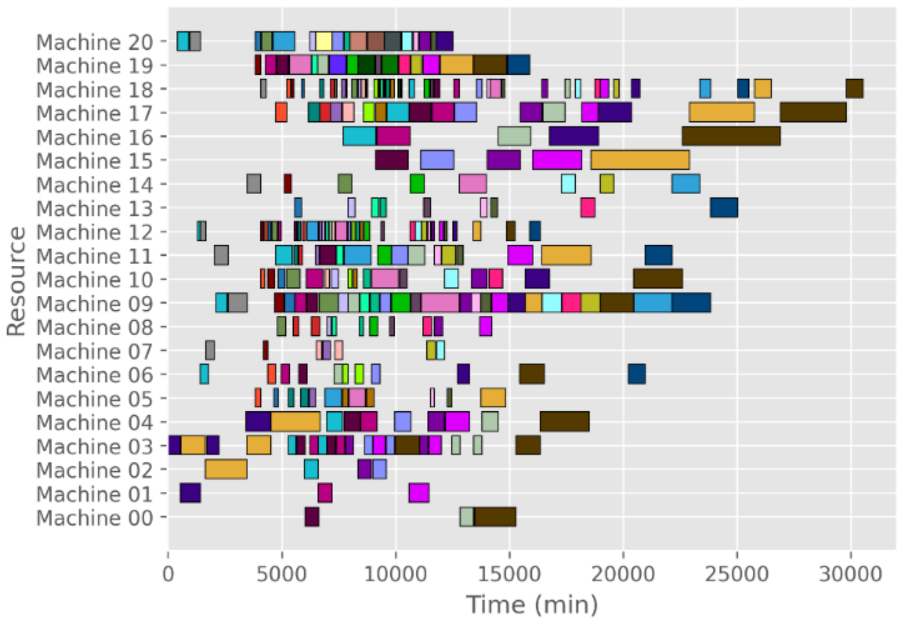

Figure 1.3: An example of complex production scheduling.

The complexity of scheduling is further amplified by the interdependencies between different tasks and processes. The production line operates as a chain, where the completion of one task triggers the start of another. Delays or disruptions at any stage can have a domino effect, leading to bottlenecks and inefficiencies down the line. Ensuring smooth flow and coordination between tasks is crucial, and even minor changes in the schedule can have significant ripple effects on the entire production process.

Moreover, real-world production environments are often subject to uncertainties and disruptions. Machine breakdowns, unexpected material shortages, fluctuating demand, and workforce issues are just a few examples of the unpredictable events that can occur. Schedulers must account for these uncertainties and be prepared to adjust the schedule in real-time to maintain productivity and meet delivery deadlines.

Another factor contributing to the difficulty of scheduling a production line is the presence of resource constraints. Limited availability of machines, workforce, or raw materials imposes limitations on the production capacity. Optimally allocating these finite resources across various products and tasks becomes a complex optimization problem.

Additionally, modern production lines are becoming increasingly dynamic and flexible to meet market demands. The concept of *lot size one* or personalized production requires rapid changeovers and frequent adjustments to the schedule. As a result, traditional static scheduling methods are no longer sufficient, and agile scheduling techniques are needed to adapt quickly to changing production requirements.

Furthermore, the objective of scheduling is not just limited to minimizing production time or costs. In today's competitive landscape, organizations also need to consider other critical factors such as on-time delivery and customer satisfaction. Consider these multiple objectives requires a comprehensive approach and the use of advanced optimization algorithms.

Finally, due to the inherent variability and diversity of products, interdependencies between tasks, uncertainties and disruptions, resource constraints, dynamic production requirements, and multiple competing objectives, scheduling a production line becomes challenging. To address these complexities, advanced scheduling techniques such as AI-based algorithms, simulation models, and optimization methods such as Mixed Integer Linear Programming (MILP) are required. By confronting these challenges head on, organizations can improve production efficiency, lower costs, increase customer satisfaction, and gain a competitive advantage in the ever-changing manufacturing and production landscape.

## 1.2 See The Future

As a basis for comparison with the solutions generated by the mathematical model here proposed, a state-of-the-art software used by manufacturing companies that generates solutions for scheduling problems was used for comparison. The software uses a processing kernel called See The Future (STF), developed by the National Institute of Technology (INT), and is made available through a customized layer for each client.

A list of the jobs that need to be completed, together with the corresponding delivery dates, dependencies between the operations of each job, machine options for completing each operation, and specific factory data like processing times, setup times between operations, production rates, and various other parameters must be provided by the customer. Then, in order to prepare the data for use by its processing core, the software processes and rearranges the data.

STF is able to generate viable solutions to a wide variety of scheduling problems using different heuristics and user guidance. At each iteration, the user selects a set of rules that will be applied in order of priority. Users can even specify these sequencing criteria and request that the STF developers implement them. In an iterative and interactive approach, the user then changes not only the priority and set of rules that the STF will use to build the solution, but can also include a variety of other soft constraints that, while not necessarily impractical, do not fit with the culture and needs of the factory in question. In this sense, the user's knowledge of the problem and the identification of bottlenecks in the process, for example, are crucial to the proper use of the STF.

Thus, STF can generate viable solutions to a wide range of scheduling problems. The common approach to identifying a good feasible solution is an iterative procedure of rearranging available rules based on the generated partial schedule. This set of rules is employed when selecting each of the possible combinations of jobs, operations, and machines (a triplet) among the jobs and operations to be assigned to the respective machines at each instant.

In other words, the STF discretely traverses the time established for production planning to decide, deterministically and greedily, which production tasks should occur at each instant. To do so, the STF cleverly and leanly explores all possible combinations of operation and machine and evaluates them according to the policy established by the user at each decision time.

To compare different allocations, different queuing policies can be used: First In, First Out (FIFO); Shortest Deadline (SD); Shortest Setup Time (SST); Shortest Processing Time (SPT), among other specific policies devised by the customer in question. Whenever two triples share the same resources within the planning horizon and can be evaluated for machine allocation (i.e., performing a given operation on a machine at a given instant is feasible), these queuing rules come into effect, being evaluated in increasing order of priority.

The triple that wins in at least one of the comparisons at each iteration becomes a processing task in the feasible solution and is not reconsidered. Once all the feasible triples become processing tasks, the simulator then steps to the next instant in time when some event occurs that performs a change in the state of some entity of the simulation (such as the completion of a processing task or the release of some machine either by termination of processing or by finished maintenance, for example). A feasible solution is presented to the user once all jobs' operations have been processed and scheduled.

Note that the STF does not have an explicit objective function. However, the user can determine different combinations of queuing rules, which are considered when evaluating the triplets to generate a feasible schedule, in order to modify the planning generated by the simulator to improve the performance metrics to be evaluated (e.g. reduce delay, reduce idleness, increase profit).

## 1.3 Contributions

The main objective of this work is to present two mathematical models that successfully handle the objectives and complexities present in real-world manufacturing environments. The main goal of this research has been to develop a model that can generate solutions that are viable and efficient in real production environments.

An approach was devised to effectively address the sophisticated dependency graph of operations associated with each item in real-world manufacturing environments. To ensure comprehensive coverage of all possible arrangements of dependencies among operations, a Directed Acyclic Graph (DAG) approach was employed in the formulation of the mathematical models. By representing the dependencies among operations of an item as a DAG, the proposed models possess the inherent capability to handle multifaceted interdependencies, thus enabling a holistic and accurate representation of the actual production processes. This representation ensures that the mathematical models presented in this research are capable of addressing a wide array of real-world production challenges, making them highly versatile and relevant in practical production environments.

To ensure the best possible alignment with real production challenges, the essential characteristics and features that must be incorporated into the suggested models were first carefully identified. It is ensured that the mathematical models proposed here have the capabilities to faithfully reproduce the key aspects of actual production processes by providing such careful consideration.

Furthermore, the research work undertakes an evaluation process involving benchmark and generated instances and real-world manufacturing scenarios. This evaluation scenario not only tests the efficacy of the proposed mathematical models but also validates their practical utility in solving real-world manufacturing problems. The empirical results derived from this evaluation strongly indicate that the proposed mathematical models yield optimal solutions for medium-sized benchmark instances, underscoring their computational efficiency and efficacy.

Additionally, the efficacy of the proposed models is further exemplified by their performance on simplified real-world instances. In comparison to a state-of-the-art tool widely employed in manufacturing processes (STF software), the solutions obtained from the proposed models exhibit a notable improvement, thus affirming their practical superiority in addressing real-world production problems.

Finally, as part of this work, an article was published in the Brazilian Symposium on Operational Research (SBPO) [23].

## 1.4 Structure

The remainder of this paper is organized as follows:

- Chapter 2 provides a review on the relevant literature about the Job Shop and Flexible Job Shop problems tackled in this work and about the Dependency Graph approach as an Directed Acyclic Graph, including how researchers have used CP and MIP models, metaheuristics and others to tackle this class of problems. An explanation about some real-world scenarios are also provided and along with gaps identified;

- Chapter 3 presents four proposed mathematical models and a detailed explanation about their objective and constraints.

- In Chapter 4, two sets of experiments are presented and discussed: the first set contains comparison results between mathematical models and a state-of-the-art software based on heuristics and discrete event computer simulation (STF); the second set provides results on using solutions generated by STF to improve the results initially obtained by mathematical models.

- Chapter 5 presents a brief conclusion of this thesis and a path to future works.

# Chapter 2

# Related work and background

The Job Shop Scheduling Problem (JSSP) is a well-known combinatorial optimization problem where the objective is usually to minimize the makespan. The JSSP has many applications in manufacturing, logistics, project management, and other domains. However, the JSSP is also known to be NP-hard, which means that finding an optimal solution may be computationally intractable for large instances. Therefore, many researchers have proposed various heuristic and metaheuristic algorithms to find near-optimal solutions in reasonable time.

However, most of the existing algorithms for the JSSP assume that the operations can start as soon as the preceding operations are finished, and that the only objective is to minimize the makespan. These assumptions may not hold in more realistic scenarios, where there may be additional constraints and objectives that need to be considered. For example, in the cases presented in this work, some jobs have a minimum start time that specifies the earliest time that a job can begin. This could be due to a lack of raw materials or even to do reschedule, so the current job at some machine can be finished. Furthermore, there may be different objective functions that reflect the decision maker's preferences or priorities, such as minimizing total tardiness, which is a common approach when some features that must be considered are related to dates.

In this dissertation, we address the JSSP under more realistic constraints and objectives, and propose novel algorithms that can handle them effectively and efficiently. We first review the relevant literature on the JSSP and its variants, and identify the research gaps and challenges that motivate our work. We then present the models proposed in Chapter 3, each focusing on a different aspect of the problem.

## 2.1 Job Shop Scheduling Problem

In computer science and operations research, the JSSP is a well-studied combinatorial, NP-hard optimization problem. The goal of the problem is to identify the best plan for distributing shared resources to competing activities over time in order to reduce the total time required to complete all activities.

In a general job scheduling problem, we are given $n$ jobs with varying processing times that must be scheduled on m machines with varying processing speed, while attempting to minimize the makespan - the total length of the schedule (that is, when all of the jobs have completed processing). Each job in the job-shop scheduling variant consists of a set of $n_j$ operations that must be executed in a specified order (known as precedence constraints) [24]. When an operation of a job is assigned to be processed on a machine it is called a task of production.

There are several constraints for the job shop problem: no operation for a job can be started until the previous operation for that job is completed. A machine can only work on one task at a time. A task, once started, must run to completion and each operation requires specific resources (e.g. machines, operators etc.) to be processed.

A common goal for this problem is to schedule the tasks on the machines with the objective to minimize the length of the schedule—the time it takes to complete all of the jobs, also known as makespan.

Another set of constraints commonly present in real production lines are related to dates. The start of production is sometimes limited to the arrival of raw materials, or it may even depend on a process that is performed in another part of the factory. This means that the demand exists, but it should not be started before a predetermined time. In the latter situation, this type of constraint can be avoided if the planning scope is expanded to cover all sectors responsible for one or more operations on a certain item [25]. However, it is not unusual to find contexts so complex that, to make the planning process feasible, it becomes necessary to break the planning of the whole plant into smaller parts, such as physical sectors or related areas.

On the other hand, deadlines for the production of a demand are also part of the planning routine in real factories. In production lines that operate both by pushed or pulled demand (which generally involves a strict inventory control), deadlines are necessary to control what to perform at each moment of planning. The difference is that in pushed production these deadlines tend to be more flexible, because the demand deadlines are intended to ensure the minimum safety levels of operations. While in pushed production, this deadline can be crucial to the organization's success, especially in cases where deadlines are set through agreements with end customers. In this case, it is clear that meeting or not meeting a deadline has immediate effects on customer relationship management (while also accruing early bonuses or paying late fees).

Other aspects of JSSP, such as the management of energy or human resources, are beyond the scope of this work. However, a relevant case study of finite capacity production scheduling in a motorcycle manufacturing plant was conducted by CARVALHO *et al.* [8]. The authors employed simulation-based systems to support production schedulers in different plant sectors. An action research method was used, which included participatory and iterative problem-solving and improvement processes. The authors took into account the human, organizational, and technological dimensions of the scheduling problem and emphasized the implementation lessons learned. The authors reported significant benefits from their approach, such as reduced work in process, assembly line stoppages, and scheduling time and also discussed the implementation challenges and difficulties, such as modeling complexity, data integration, user training, and organizational change.

In order to improve the modeling for the problem, the order of precedence between work operations can be treated as a DAG in this work and will be explained in section 2.3.

## 2.2 Flexible Job Shop Scheduling Problem

The Flexible Job Shop Scheduling Problem (FJSP) is an extension of the conventional job shop scheduling problem in which an operation can be executed by any machine from a defined set.

In contrast to the conventional job shop scheduling problem, where each operation must be processed on a specific machine, the FJSP allows for more flexibility in assigning operations to machines. The FJSP is far more complex than the JSSP because it introduces the assignment of operations to machines as an additional decision variable.

A mathematical model is considered efficient and effectively used for scheduling in production lines, when its constraints must reflect the characteristics of the real world whose model it is intended to represent [26]. In this section the relevant characteristics commonly found in the context of FJSP production lines are presented.

One of the main characteristics of FJSP is the need to set up a machine to start producing an item after it has been used to produce another kind of item. Often this set up time depends on both kinds of items: produced before and to be produced [27], and is called sequence dependent setup time. It is well known that setup times have a major impact on scheduling performance [28]. This effect stems from the need for machine adjustments for different reasons, such as the difference in dimensions of the items produced, different production temperatures (which implies heating or cooling of the machine), or even the change of large tools to be used (e.g. molds).

The FJSP models have become more complex over time, which motivates some researchers to review the state-of-the-art methods and identify the research gaps and directions in this field. XIE *et al.* [10] present a systematic literature review of the solution methods for the FJSP, which they divide into three main categories: exact algorithms, heuristics and meta-heuristics. They also discuss the real-world applications of the FJSP in various industrial sectors, such as glass, textile, steel and electronics. Moreover, they explore the future research challenges and opportunities of the FJSP, such as incorporating environmental factors, uncertainty factors, multi-objective optimization and hybrid models. They argue that the FJSP is a worthwhile and important research topic that requires more attention from both academia and industry.

The evolution of industry technologies and their interconnections have increased the complexities of FJSP. ZHENG *et al.* [1] performed a systematic literature review on how Industry 4.0 enabling technologies are applied in manufacturing business processes. They discovered that production scheduling and control was the most studied process, while IoT, Big Data Analytics and Cloud were the most utilized technologies. They also discussed how various technologies can be integrated to enhance different processes in the supply chain, such as production, distribution, and consumption. They also identified some of the emerging trends that are transforming the supply chain landscape, such as the shift from product-based to service-based models and the adoption of circular supply chain management principles that aim to reduce waste and environmental impact.

KIM e BOBROWSKI [28] investigated how sequence-dependent setup time affects job shop scheduling performance. They used a simulation model of a nine-machine job shop with six different job types and varying setup times for each job type. They compared four scheduling heuristics that took setup time and/or due date information into account when making a sequencing decision. They also varied the tightness of the due date and the cost structure to test the heuristics' sensitivity. Their findings revealed that setup-oriented heuristics, which explicitly accounted for setup time, outperformed ordinary heuristics in terms of flow time, due date performance, and total cost. They also discovered that setup-oriented heuristics were less sensitive to changes in due date tightness and setup time than conventional heuristics. Their research found that sequence-dependent setup time has a significant impact on job shop scheduling performance and that appropriate scheduling heuristics are required to effectively address this issue.

Mathematical models that represent the increasingly complex reality of factories are still much needed, and for this reason, studies have been carried out with the objective of improving the performance and comprehensiveness of the models over the years.

A MILP model and a hybrid metaheuristic (hybridization of Artificial Immune and Simulated Annealing Algorithms) for FJSP is introduced by ROSHANAEI *et al.* [29]. Although they have addressed the basic constraints (e.g. Sequence Dependent Setup Times (SDST)) of the FJSP with excellence, bringing a structurally efficient formulation, they do not take into account real constraints of this context such as deadlines, startup constraints and, more importantly, they do not address problems where a machine needs to be set up between two operations of the same job.

The survey provided by CHAUDHRY e KHAN [26] covers techniques that have been used to tackle FJSP. Among the techniques used in the reviewed papers are metaheuristics such as Ant-Colony Optimization (ACO), Evolutionary Algorithms (EAs), heuristics, exact algorithms such as MIP, and hybrid algorithms using one or more heuristics and/or metaheuristics.

YEUNG *et al.* [30] investigated a supply chain scheduling problem involving a single supplier, a single manufacturer, and multiple retailers, where the manufacturer has limited production capacity and can only accept some of the retailers' orders. The problem is formulated as a two-machine proportionate flow shop scheduling problem with common due windows, with the goal of maximizing the manufacturer's profit, which is dependent on storage time, storage quantity, order sequence dependent weighted storage costs, and order idle time. The issue is similar to the job shop issue, but with different machine speeds and common due dates. To solve the problem optimally, the authors devised a pseudo-polynomial dynamic programming algorithm. They also applied their model to a real-world case study in the apparel manufacturing industry.

DEMIR e İŞLEYEN [15] reviews and compares five different mathematical formulations for FJSP, based on the type of binary variable that they use to capture the sequencing decision: sequence-position, precedence, and time-indexed variables. They compare the models based on the objective function value, CPU time, the number of variables and constraints, and the number of nonlinear entries. They evaluate the models' performance using randomly generated test problems from FATTAHI *et al.* [31]. In terms of solution quality and computational efficiency, they discover that the precedence variable-based model outperforms the others. They also discover that the time-indexed model is sensitive to the time interval chosen and necessitates a large number of variables and constraints. They conclude that mathematical programming formulations can be used to better understand the structure of FJSP and develop effective heuristics.

A MILP formulation is presented by [27] to solve FJSP with SDST. To further improve the solution quality and efficiency, the article proposes a tabu search algorithm with specific neighborhood functions and a diversification structure, which begins with a randomly generated initial solution, then applies feasibility checks and iterative searches for better solutions by employing local search procedures and making minor changes to the current solution. They also emphasize the importance of considering setup times in manufacturing line problems because setup times can have a significant impact on overall manufacturing efficiency and productivity.

An integer non-linear model is introduced by [32] to deal with FJSP-SDST, alongside by a real-world case of a manufacturer in the automotive sector producing injection molded parts. What makes this work interesting is that it takes into account other factors such as release dates and deadlines. By including release dates and deadlines in the mathematical model, the proposed approach can better reflect the constraints and objectives of a practical manufacturing environment.

A Mixed Integer Non-Linear Programming (MINLP) model for FJSP with variable batches is proposed by [33], that also takes into account a transfer time between operations processed on different machines. In their work, jobs can be divided into sub-batches so that they can be processed on different machines at the same time. This is a significant improvement over traditional FJSP models, in which each operation of a job must be completed on a single machine before proceeding to the next.

Four MILP models and a constraint programming (CP) model to solve the distributed flexible work shop scheduling problem was proposed by MENG *et al.* [20]. The MILP models are formulated based on four different modeling ideas (sequence-based, position-based, time-indexed, and adjacent sequence-based), and the CP model is formulated based on range variables and domain filtering algorithms. The experimental results showed that the sequence-based MILP model was the most efficient of the MILP models, while the CP model was very effective in finding good quality solutions for both small and large-sized problems. The CP model outperforms state-of-the-art algorithms and gets new better solutions to 11 reference instances. In addition, the best MILP model and the CP model prove the optimality of 62 best known solutions.

A MILP and two heuristics for FJSP with the aim of minimizing the makespan was developed by FATTAHI *et al.* [31]. The mathematical model was a MILP that solved the assignment and sequencing sub-problems in an integrated or hierarchical way. The heuristics were based on the Simulated Annealing (SA) and Tabu Search (TS) metaheuristics, and they also used six different search structures to combine the sub-problems. They evaluated their methods using the branch and bound technique for small problems, and the upper and lower bounds for medium and large problems. They did not consider setups, due dates, another objective function, or DAGs for modeling operations precedences, but they provided some benchmark instances that are widely used in the literature. They found that the hierarchical approach performed better than the integrated one.

A CP and MILP models were employed by ZHANG e WANG [19] to address the flexible assembly JSSP in a dynamic manufacturing environment. In addition, several dispatch criteria with machine feedback mechanism are developed. Experimental studies are conducted based on test case problems with different scales and complexities. It was found that CP is the most effective approach, whose solution aptitude outweighs MILP, as well as all dispatch criteria in both static and dynamic cases. On the other hand, the dispatch criteria are simple to implement, among which the earliest completion time criterion is the most favorable. A real-time scheduling/reprogramming system was built for the implementation of the proposed approaches to solving practical production problems.

To minimize the total energy usage, MENG *et al.* [11] present six MILP formulations for the FJSP that aim. The formulations are based on two different approaches: idle time variable and idle energy variable. The former approach computes the idle energy consumption by multiplying the idle time by the idle power, while the latter approach directly assigns idle energy as a variable to be optimized. The formulations also incorporate the switching on/off strategy to reduce the idle energy consumption when the machines are not in use for long durations. A modeling concept is used to split each machine into several slots or positions and allocate operations to these slots. A comparison and an evaluation of the proposed formulations and a previous formulation was made in terms of size and computational difficulties. The paper finds that all of the proposed formulations significantly outperform the previous formulation, and that the formulations based on different approaches have large differences in size and computational difficulties.

ÖZGÜVEN *et al.* [34] propose two MILP models for the flexible job shop scheduling problem with routing and process plan flexibility. The first model is a MILP model that employs a binary variable to indicate whether a job is processed before another one on a machine, while the second model is an extension of the first one that incorporates alternative process plans for each job. They also evaluate their models on hypothetical test instances and compare them with existing models and heuristic methods from the literature. They show that their models can obtain optimal or near-optimal solutions within reasonable CPU time. They also claim that their models have fewer binary variables and constraints than previous models.

A CP model and a MILP model to solve a JSSP with parallel batch processing machines was proposed by HAM e CAKICI [13]. They also compare their models with a previous MIP model from the literature. They test their models on a set of common problem instances and find that CP outperforms MIP in terms of computational time and solution quality. The authors emphasized that the proposed three-indexed model has fewer variables and constraints than the existing four-indexed model in the literature. In order to tighten the formulation and improve the lower bounds, they also introduce some valid inequalities.

It is presented by GOMES *et al.* [22] a MILP model for scheduling make-to-order job shops that are flexible. It takes into account groups of parallel homogeneous machines, a limited number of intermediate buffers, and negligible set-up effects. The model also supports re-circulation, which means that certain jobs may visit the same machine group multiple times. The authors compare their model to other MILP formulations and heuristic methods in the literature and demonstrate that using commercial MILP software, their model can obtain optimal solutions for realistic examples of flexible job shops.

JUNG *et al.* [35] presented a MILP model and three genetic algorithms (GA) for a two-stage assembly flow shop scheduling problem with dynamic component-sizes and a setup time. The problem involves producing various types of components on a single machine and assembling them into products on a single assembly machine. The authors considered the case where different products require different numbers and types of components, and a setup time is incurred when the machining machine switches between different components. The objective is to minimize the makespan of the production process. The authors derived several optimal properties for the problem and proposed three GAs with different chromosome representations: one with a complete solution, one with a component-manufacturing sequence, and one with a product-assembly sequence. The authors compared the performance of the GAs with randomly generated examples and found that the GA with a product-assembly sequence outperformed the other two GAs and the MILP model in terms of solution quality and computational efficiency.

An improved hybrid algorithm framework for the job-shop scheduling problem that combines MILP and CP was proposed by REN e TANG [14]. Based on the concepts of critical job assignment and critical cut, the authors show two heuristics for generating more effective cuts. Experiments show that the improved framework is capable of solving more complex problems faster than the existing framework. The paper also introduces a hybrid MILP/CP model for the job-shop scheduling problem, which divides the problem into two subproblems: assignment and sequencing. The paper builds on previous work that uses a hybrid MILP/CP approach to solve scheduling and combinatorial optimization problems.

A MILP model and a multi-objective evolutionary algorithm based method for solving the dynamic flexible job shop scheduling problem was introduced by SHEN e YAO [17] . They considered four objectives: make-span, tardiness, maximal machine workload, and stability. They proposed a modified predictive–reactive scheduling approach that used heuristic strategies to construct the initial population and problem-specific genetic operators for variation. They also designed a dynamic decision making procedure to select one solution from the Pareto front. They compared their method with existing scheduling rules and static algorithms on a simulated dynamic flexible job shop.

VITAL SOTO [18] developed a MILP model and a hybrid bacterial foraging optimization algorithm HBFOA are proposed to tackle the the FJSP with sequencing flexibility (FJSPS), which is a generalization of the classical job-shop problem that allows different precedence relations among operations. They also incorporate dual-resource constraints (workers and machines) and multiple process plans for each job into the problem. They formulate multi-objective mathematical models and a modified non-dominated sorting genetic algorithm to deal with these extensions. The authors evaluate their models and algorithms on various benchmark instances and a real-world case study from a metal cutting industry. They demonstrate that their methods can generate efficient and effective solutions for the FJSPS and its extensions [36].

A FJSP with job-splitting was studied by TUTUMLU e SARAÇ [37]. They developed a MILP model that enables the jobs to be divided into variable sub-lots and allocated to different parallel machines without any restrictions on the number and size of sub-lots. They also designed a hybrid genetic algorithm that uses a local search algorithm to optimize the size of sub-lots and enhance the search performance. Their goal was to minimize the makespan. They evaluated their model and algorithm on randomly generated problems of various sizes and found that job-splitting can significantly decrease the makespan. They also showed that their hybrid genetic algorithm outperforms a classical genetic algorithm.

All the related works presented have some aspects of real-world problems identified in this paper, although there is a gap, which this paper aims to fill, when it comes to dealing with all aspects together.

A further distinction between this work and the aforementioned works is that they approach delay in a way that considers negative values, which means the job is finishing earlier, and since, where this effect occur, there difference in the objective function from finishing an operation that is already within the agreed deadline or one that is outside is the same.

Also, some recent studies have explored the impact of Industry 4.0 on production scheduling, as well as the current and future research directions in this field.

For instance, PARENTE *et al.* [9] present a review and analysis of the impact of Industry 4.0 (I4.0) on production scheduling, as well as the current and future research directions in this field. The authors identify the main opportunities and challenges brought by I4.0 to scheduling, such as increased flexibility, adaptability, integration, and complexity. They also propose a set of critical scheduling areas (CSA) that need to be developed to enable the full implementation of I4.0 in scheduling, such as holistic scheduling, decentralized decision-making, human–robot collaboration, and scheduling under uncertainty. The authors conduct a two-stage cascade literature review to assess the state of the art and the research gaps in each CSA and provide suggestions for future research.

An overview of the evolution of production paradigms from mass production to mass personalization is presented by WANG *et al.* [2], with a discussion of the role of Industry 4.0 in enabling the latter. The authors propose a framework for mass personalization production based on four key components of Industry 4.0: cyber-physical system, mobile and cloud computing and Internet of things, big data, data mining and knowledge discovery, and Internet of service. They also provide some examples of industrial practices and a lab demonstration of producing personalized keychains using the proposed framework. The authors conclude that mass personalization is the advanced stage of mass customization, and that Industry 4.0 can facilitate the realization of mass personalization by integrating emerging technologies and customer participation in the design process.

A survey on heuristic approaches for solving Multi-Objective Flexible Job Shop Problems (MOFJSP) was conducted by TÜRKYILMAZ *et al.* [21] , which are complex optimization problems with wide applications in industrial fields. The survey reviewed the recent studies on MOFJSP based on different heuristic techniques, such as particle swarm optimization, artificial bee colony, variable neighborhood search, tabu search, and others. The survey also analyzed the problem sets, objective functions, and performance measures used in the literature. The survey concluded that there is no general best method for solving MOFJSP, and that more research is needed to address the challenges and limitations of the existing methods.

JIANG *et al.* [7] present a comprehensive review of the evolution of production scheduling from Industry 3.0 to Industry 4.0, covering different types of scheduling problems, methods, and applications. They classify scheduling problems into four categories: centralized, distributed, decentralized, and cloud manufacturing scheduling. They also analyze the literature on these topics from various perspectives, such as production system configuration, operating environment, objectives and constraints, and solution methods. They use both heuristic and mathematical models to solve scheduling problems, depending on the complexity and characteristics of the problems. They also discuss the challenges and trends in production scheduling in the context of Industry 4.0, such as sustainable manufacturing, mass customization, inter- and intra-organizational collaboration, adaptive and self-organized scheduling, and big-data driven scheduling. They conclude by highlighting the importance of production scheduling for improving the efficiency and intelligence of manufacturing systems.

A structured literature review of large scale industrial production scheduling problems with complex constraints was proposed by SCHLENKRICH e PARRAGH [38], they classify the existing literature into two streams: one that generalizes classical scheduling problems and one that focuses on specific industry cases. They identify three main categories of solution methods for large scale problems: metaheuristics, constraint programming and machine learning. They highlight the challenges and opportunities of scheduling in the era of Industry 4.0 and suggest some promising directions for future research.

Finally, VERDERAME *et al.* [39] provide a review of the planning and scheduling problems under uncertainty across multiple sectors, such as chemical, petrochemical, pharmaceutical, energy, power, agriculture, forestry, waste, water, transportation, and others. They compare the commonalities and differences of the problem formulations, objectives, constraints, and uncertainty approaches used in each sector. They also identify the key challenges and opportunities for future research in this area. They conclude that planning and scheduling under uncertainty is a rich and diverse field that can benefit from interdisciplinary collaborations and integrated methodologies.

Table 2.1 summarizes the characteristics of some of the papers examined that deal with the JSSP and the FJSP, listing the features of each approach presented making it easier to see how they compare.

Table 2.1: Summary of the papers' reviewed with the characteristics of approaches used to tackle JSSP/FJSP.

| **Paper** | **MILP** | **MINLP** | **Flexible** | **Heur** | **Setup** | **Dates** | **DAGs** | **Objective Function** |
|---|---|---|---|---|---|---|---|---|
| BIRGIN *et al.* [12] | ✓ | - | ✓ | - | - | - | ✓ | Makespan |
| CAO *et al.* [40] | - | - | ✓ | ✓ | ✓ | - | ✓ | Makespan |
| CARVALHO *et al.* [8] | - | - | ✓ | ✓ | ✓ | ✓ | - | Multi-objective |
| FATTAHI *et al.* [31] | ✓ | - | ✓ | ✓ | - | - | - | Makespan |
| GOMES *et al.* [22] | ✓ | - | ✓ | - | - | ✓ | - | Cost |
| HAM e CAKICI [13] | ✓ | - | ✓ | - | ✓ | - | - | Makespan |
| JUNG *et al.* [35] | ✓ | - | - | - | ✓ | - | - | Makespan |
| MENG *et al.* [11] | ✓ | - | ✓ | - | - | - | - | Energy |
| MENG *et al.* [20] | ✓ | - | ✓ | - | - | - | - | Makespan |
| ÖZGÜVEN *et al.* [34] | ✓ | - | ✓ | - | - | - | - | Makespan |
| PINHA *et al.* [25] | - | - | ✓ | ✓ | ✓ | ✓ | - | Multi-objective |
| REN e TANG [14] | ✓ | - | - | ✓ | ✓ | ✓ | - | Makespan |
| ROSHANAEI *et al.* [29] | ✓ | - | ✓ | ✓ | - | - | - | Makespan |
| SHEN e YAO [17] | ✓ | - | ✓ | ✓ | ✓ | ✓ | - | Efficiency |
| SHEN *et al.* [27] | ✓ | - | ✓ | ✓ | ✓ | - | - | Makespan |
| TUTUMLU e SARAÇ [37] | ✓ | - | ✓ | ✓ | ✓ | - | - | Makespan |
| VITAL SOTO [18] | ✓ | - | ✓ | ✓ | - | ✓ | ✓ | Tardiness |
| WINKLEHNER e HAUDER [32] | - | ✓ | ✓ | ✓ | ✓ | ✓ | - | Makespan |
| XIULI *et al.* [33] | ✓ | ✓ | ✓ | ✓ | - | - | - | Makespan |
| YEUNG *et al.* [30] | - | - | ✓ | ✓ | - | ✓ | - | Profit |
| ZHANG e WANG [19] | ✓ | - | ✓ | - | ✓ | - | - | Makespan |

## 2.3 Dependency graph

A directed acyclic graph (DAG) is a modeling approach that can be used in the context of the FJSP to encode the operation dependencies between different jobs and machines, providing a powerful and intuitive representation of the scheduling constraints. The classic linear order of precedence can be seen in Figure 2.1, and is also a DAG.

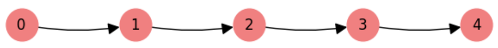

Figure 2.1: An example of job with a linear order of precedence.

A DAG is a specialized type of graph where the edges have a specific direction, meaning they go from one node to another, indicating the flow or precedence of operations. In the job shop context, each node in the DAG corresponds to an operation that needs to be performed, and the directed edges represent the order in which operations must be executed, indicating the precedence constraints between operations. The acyclic property ensures that no circular dependencies exist, ensuring a feasible sequence of operations in the job shop.

The DAG approach provides a compact and informative way to visualize the scheduling constraints and explore the feasible sequence of operations for each job. The use of DAG approach overrule the traditional representation of FJSP where only one operation precedes another, increasing complexity and bringing the representation closer to the real ones in the manufacturing context. In this work, two types of DAGs were investigated in order to address some common scenarios in production planning, as shown below.

A MILP-based method for the FJSP problem was proposed by BIRGIN *et al.* [12], who improved an existing model by ÖZGÜVEN *et al.* [34] . The proposed method uses DAGs to represent the jobs and was tested using both exact and heuristic methods. The work results showed better performance and flexibility than the previous model. The authors also generated new test cases for the FJSP problem, based on realistic scenarios from the printing industry.

A approach to FJSP was proposed by CAO *et al.* [40], which combines a knowledge-based cuckoo search algorithm (KCSA) and Reinforcement Learning (RL). The KCSA can handle complex scheduling constraints such as mating operations, sequence-dependent setup time, and precedence constraints, which are modeled as directed acyclic graphs (DAGs), enhancing the representations' flexibility. The algorithm enhances a basic cuckoo search algorithm with reinforcement learning and hybrid heuristics, forming a knowledge base that influences mutation and adjusts parameters dynamically. A feedback mechanism is also implemented to ensure the appropriate balance between population diversity and intensification. The

KCSA's effectiveness and robustness were verified using benchmark and randomly generated cases, showing superior performance when compared to other methods.

VITAL-SOTO *et al.* [36] also used DAGs to represent operations precedence relations rather than in a linear order, but without specifying which class of DAGs were used or whether a procedure was used to choose the precedence constraints rather than a random assignment in evaluating different scenarios.

Despite the fact that some works have employed DAGs to represent operations precedence relations [12, 36, 40], this work extends the use of DAGs to model operations precedence relations in two different ways, each suited for a different production scenario:

- Barabási-Albert Graphs

  Barabási and Albert [41] proposed random graph models to understand how complex networks emerge and over time. The Barabási-Albert (BA) graphs have a property called scale-free which means that the number of connections per node follows a power law distribution. This implies that there are a few nodes, called hubs, that have many connections, while most nodes have only a few. Random graphs can model various phenomena, such as the structure of the internet, social networks, or biological systems, and play a crucial role in information dissemination, robustness, and overall network efficiency [42]. In job shop problems, a DAG can be likened to a BA graph in representing dependencies between different operations. By representing the operation dependency graph as a Barabási-Albert-like DAG, the characteristics of different operations precedence arrangements can be explored, providing a more realistic depiction of job shop problems and evaluating scheduling algorithms' efficiency in complex and interconnected environments. Figure 2.2 shows an example of BA graph.

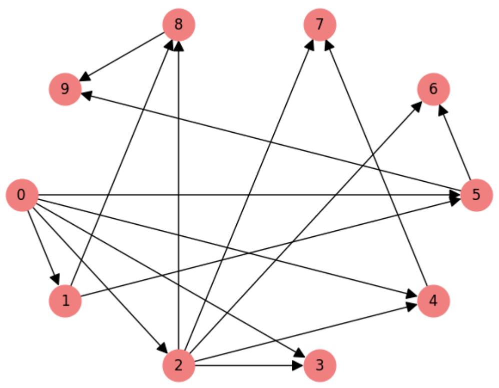

Figure 2.2: An example of a job with a precedence such as a BA graph.

- Uniform Random Trees Graphs

A Uniform Random Tree (URT), which is a tree-like structure with some special properties, consists of nodes, each of which has a unique parent and an arbitrary number of children. The parent-child relationships are randomly assigned, so that there is no fixed pattern or order in the URT. This makes the URT a versatile and flexible way of representing scheduling constraints, as it can capture dependencies among operations in a job. The URT graph's complexity and degree behavior are both intriguing, since its complexity is determined by the number of nodes in the tree, which can range from one to infinity. A node's degree is determined by the number of children it has, which can range from zero to infinity. A URT has an average degree of one (children), but the distribution of degrees is not constant with some nodes having zero or one child and a few nodes having many children. This reflects the URT structure's randomness and diversity. Figure 2.3 shows an example of URT graph.

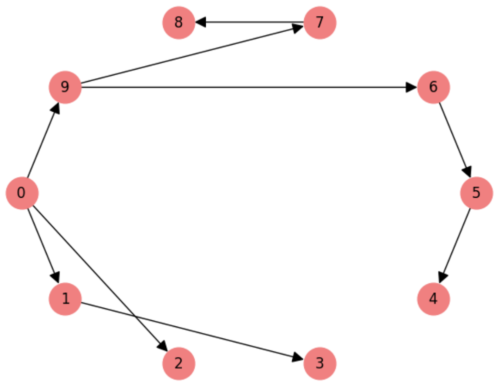

Figure 2.3: An example of a job with a precedence such as a URT graph.

Finally, the URT graphs can capture the assembly operations that are common in some situations, while the BA graphs can handle the complex dependencies among many items that are required for a final product. This work provides a deeper understanding of how DAGs can be used to represent different production contexts.

# Chapter 3

# Mathematical models

This chapter presents the mathematical models proposed to solve instances of FJSP. The proposed models assume that start times, processing times, and setup times are integers, and that time intervals smaller than seconds are negligible, allowing time intervals to be discrete and measured in minutes or hours. Four models are proposed and can be classified in two ways: by the type of the objective function and by the type of precedence of jobs' operations. Table 3.1 shows how models are divided by each classification.

| | | Objective | |
|---|---|---|---|
| | | Makespan | Deadline |
| Prec. Type | Linear | Model-1 | Model-2 |
| | DAGs | Model-3 | Model-4 |

Table 3.1: Table containing the classification by objective funcion and precedence relationship type of the mathematical models proposed.

The models' indices and parameters are described in Table 3.2, while the decision variables are described on Table 3.3 for all models proposed.

Table 3.2: Indexes and parameters.

| **Notation** | **Description** |
|---|---|
| $n$ | Number of jobs. |
| $m$ | Number of machines. |
| $i$ | Index for machines, where $1 \leq i \leq m$. |
| $j, h$ | Indices for jobs, where $1 \leq j, h \leq n$. |
| $n_j, n_h$ | Number of operations of jobs $j$ and $h$. |
| $l, k$ | Indices for operations of a job, where $1 \leq l \leq n_j$, $1 \leq k \leq n_h$. |
| $M$ | A large positive number. |
| $R_{j,l}$ | The set indicating the machines eligible to process operation $l$ of job $j$. |
| $P_{i,j,l}$ | Processing time of $l$th operation of job $j$ on machine $i$. |
| $O_{i,j,l,h,k}$ | Sequence dependent setup times between $l$th operation of job $j$ that precedes $k$th operation of job $h$ on machine $i$. |
| $Q_j$ | Minimum starting time for job $j$. |
| $D_j$ | Deadline for job $j$. |
| $U_{j,l}$ | Indices of operations of job $j$ that must succeed operation $l$ of the same job |

Table 3.3: Decision variables.

| **Notation** | **Description** |
|---|---|
| $y_{i,j,l}$ | Binary decision variable indicating whether job $j$ is processed in machine $i$ in stage $l$. |
| $x_{j,l,h,k}$ | Binary decision variable indicating whether $l$th operation of job $j$ is processed after $k$th operation of job $h$ in any machine. |
| $s_{i,j,l}$ | Continuous variable for starting time of operation $l$ of job $j$ on machine $i$. |
| $b_{i,j,n_j}$ | Auxiliary variable indicating whether the completion of a job is greater than its deadline. |

## 3.1 Model-1

The first mathematical model (Model-1) proposed in this work to solve more realistic instances of the FJSP is a MILP that has both classic constraints, commonly found in mathematical modeling papers on FJSP, and additional constraints, that better represent real production environments. While these constraints are not new, they are usually found separately, in different works, and under different modeling perspectives. Thus, the proposed model consolidates prior ideas into a single and more realistic mathematical model.

The proposed MILP model is presented as follows:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l:U_{j,l}=\emptyset} (s_{i,j,l} + P_{i,j,l} \cdot y_{i,j,l}) \tag{3.1}$$

Subject to:

$$\sum_{i=1}^{m} s_{i,j,l+1} \geq \sum_{i=1}^{m} s_{i,j,l} + \sum_{i=1}^{m} P_{i,j,l}\, y_{i,j,l} \qquad \forall j,\ \forall l < n_j \tag{3.2}$$

$$\begin{aligned} s_{i,j,l+1} \geq s_{i,j,l} + P_{i,j,l} \cdot y_{i,j,l} + O_{i,j,l,j,l+1} - M(2 - y_{i,j,l+1} - y_{i,j,l}) \\ \forall i \in \{R_{j,l} \cap R_{j,l+1}\},\ \forall j,\ \forall l < n_j \end{aligned} \tag{3.3}$$

$$\begin{aligned} s_{i,j,l} \geq s_{i,h,k} + P_{i,h,k} + O_{i,h,k,j,l} - M(3 - x_{j,l,h,k} - y_{i,j,l} - y_{i,h,k}) \\ \forall i \in \{R_{j,l} \cap R_{h,k}\},\ \forall j < n,\ \forall l,\ \forall h > j,\ \forall k \end{aligned} \tag{3.4}$$

$$\begin{aligned} s_{i,h,k} \geq s_{i,j,l} + P_{i,j,l} + O_{i,j,l,h,k} - M(x_{j,l,h,k} + 2 - y_{i,j,l} - y_{i,h,k}) \\ \forall i \in \{R_{j,l} \cap R_{h,k}\},\ \forall j < n,\ \forall l,\ \forall h > j,\ \forall k \end{aligned} \tag{3.5}$$

$$s_{i,j,1} \geq Q_j \cdot y_{i,j,1} \qquad \forall i,\ \forall j \tag{3.6}$$

$$\sum_{i=1}^{m} y_{i,j,l} = 1 \qquad \forall j,\ \forall l \tag{3.7}$$

$$s_{i,j,l} \leq M \cdot y_{i,j,l} \qquad \forall i,\ \forall j,\ \forall l \tag{3.8}$$

$$s_{i,j,l} \geq 0 \qquad \forall i,\ \forall j,\ \forall l \tag{3.9}$$

$$y_{i,j,l} \in \{0,1\} \qquad \forall i,\ \forall j,\ \forall l \tag{3.10}$$

$$x_{j,l,h,k} \in \{0,1\} \qquad \forall j < n,\ \forall l,\ \forall h > j,\ \forall k \tag{3.11}$$

The Objective Function (3.1) aims to minimize the makespan, and it is defined by the sum of the finishing time of the all operations with no successors of all jobs. Thus, it might be possible to reduce the sum by decreasing the finish time of each job.

Constraints (3.2) relates the start times of subsequent operations of the same job, establishing that a successor operation $l+1$ of job $j$ on machine $i$ can only begin when its predecessor operation $l$ on the same machine has started and finished. Constraints (3.3) work similarly, ensuring that the setup times of subsequent operations of the same job are respected. In fact, whenever $y_{i,j,l} = 1$ and $y_{i,j,l+1} = 1$, then the right-hand-sides of (3.3) are tight, and (3.3) are active.

Constraints (3.4) and (3.5) act together, and deal with the precedence of two operations ($l$ and $k$) from jobs ($j$ and $h$), that are performed on the same machine. Necessarily, one of the operations must occur before the other, since it is not allowed for a machine to process two operations at the same time. Therefore, for these constraints to come into play, at least $y_{i,j,l}$ and $y_{i,h,k}$ must be equal to 1. Thus, whenever the $l$th operation of job $j$ is processed after $k$th operation of job $h$ in some machine ($x_{j,l,h,k} = 1$), then the right-hand-sides of (3.4) are tight, ensuring that the precedence between the operations is respected. When $x_{j,l,h,k} = 0$, Constraints (3.5) guarantee the same effect, but in the reverse order of precedence. This way, the setup time between two subsequent operations is respected, since this time is added as well as the processing time for that operation on that particular machine.

Constraints (3.6) couple variables $s$ and $y$ with parameter $Q$, ensuring that the first operation ($l = 1$) of job $j$ assigned to machine $i$ starts at a minimum time $Q_j$. Thus, $s_{i,j,1} \geq Q_j$ whenever $y_{i,j,1} = 1$.

Constraints (3.7) guarantee that each machine only processes one job at a time. Constraints (3.8) couple variables $s$ and $y$, ensuring that an operation $l$ of job $j$ starts on machine $i$ only if job $j$ is processed in machine $i$ in stage $l$. Thus, if $y_{i,j,l} = 0$ then $s_{i,j,l} = 0$.

Lastly, Constraints (3.9), (3.10) and (3.11) determine the decision variables' domains.

## 3.2 Model-2

Another common objective in production line sequencing is meeting deadlines or minimizing tardiness. This alternative MINLP model is represented as follows:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}(s_{i,j,n_j} + P_{i,j,n_j} - D_j) \cdot b_{i,j,n_j} \tag{3.12}$$

Subject to (3.2)-(3.11) and:

$$s_{i,j,n_j} + P_{i,j,n_j} \cdot y_{i,j,n_j} \geq D_j - M \cdot (1 - b_{i,j,n_j}) \qquad \forall i,\ \forall j \tag{3.13}$$

$$s_{i,j,n_j} + P_{i,j,n_j} \cdot y_{i,j,n_j} \leq D_j + M \cdot b_{i,j,n_j} \qquad \forall i,\ \forall j \tag{3.14}$$

$$b_{i,j,n_j} \in \{0, 1\} \qquad \forall i,\ \forall j \tag{3.15}$$

Remembering that auxiliary variables $b_{i,j,n_j}$ indicate whether the completion of a job is greater than its deadline, Constraints (3.13) and (3.14) act together, capturing the occurrence of a delay in some job $j$. Whenever $s_{i,j,n_j} + P_{i,j,n_j} \cdot y_{i,j,n_j} < D_j$, then $b_{i,j,n_j} = 0$; otherwise Constraints (3.13) would imply $s_{i,j,n_j} + P_{i,j,n_j} \cdot y_{i,j,n_j} \geq D_j$, a contradiction. Therefore, $b_{i,j,n_j} = 0$, Constraints (3.14) are active (tight), Constraints (3.13) are non-active, and no delay is accounted for in the OF. Conversely, whenever $s_{i,j,n_j} + P_{i,j,n_j} \cdot y_{i,j,n_j} > D_j$, then $b_{i,j,n_j} = 1$; otherwise Constraints (3.14) would imply $s_{i,j,n_j} + P_{i,j,n_j} \cdot y_{i,j,n_j} \leq D_j$, a contradiction. Therefore, $b_{i,j,n_j} = 1$, Constraints (3.13) are active (tight), Constraints (3.14) are non-active, and the corresponding delay is accounted for in the OF. In the special case $s_{i,j,n_j} + P_{i,j,n_j} \cdot y_{i,j,n_j} = D_j$, we could have either $b_{i,j,n_j} = 0$ or $b_{i,j,n_j} = 1$. As the objective function is to minimize, the solver always choose $b_{i,j,n_j} = 0$, and no delay is accounted for in the OF. Finally, Constraints (3.15) determine the domain of $b_{i,j,n_j}$ variables. It is worth mentioning that (3.12) is the sum of the difference between the finish time of a job and its deadline across all jobs. Therefore, it might be possible to reduce the value of this function by delaying a job aiming to finish others earlier.

## 3.3 Model-3 and Model-4

As an advance to prior models presented, DAGs were used to represent the precedence relationship between operations of the same job. Model-3 is a variations of Model-1 and adjustments must be done in Constraints (3.2), (3.4), (3.5) and (3.6). So, Model-3 OF is shown in (3.1) and it is subject to (3.7)-(3.11) and:

$$\sum_{i=1}^{m} s_{i,j,h} \geq \sum_{i=1}^{m} s_{i,j,l} + \sum_{i=1}^{m} P_{i,j,l} \cdot y_{i,j,l} \qquad \forall j,\ \forall l,\ \forall h \in \{U_{j,l}\} \tag{3.16}$$

$$\begin{aligned} s_{i,j,l} \geq s_{i,h,k} + P_{i,h,k} + O_{i,h,k,j,l} - M(3 - x_{j,l,h,k} - y_{i,j,l} - y_{i,h,k}) \\ \forall i \in \{R_{j,l} \cap R_{h,k}\},\ \forall j < n,\ \forall l,\ \forall h \geq j,\ \forall k \end{aligned} \tag{3.17}$$

$$s_{i,h,k} \geq s_{i,j,l} + P_{i,j,l} + O_{i,j,l,h,k} - M(x_{j,l,h,k} + 2 - y_{i,j,l} - y_{i,h,k}) \quad \forall i \in \{R_{j,l} \cap R_{h,k}\},\ \forall j < n,\ \forall l,\ \forall h \geq j,\ \forall k \tag{3.18}$$

$$s_{i,j,1} \geq Q_j \cdot y_{i,j,l} \qquad \forall i,\ \forall j,\ \forall l \notin U_j \tag{3.19}$$

Constraints (3.16) relates the start times of subsequent operations of the same job, establishing that a successor operation $h$ of job $j$ can only begin when its predecessor operation $l$ has started and finished. Note that this set of constraints models the precedence constraints as DAGs and not only allows for a wide range of precedence configurations between operations in a job, but it also opens up the possibility of having more than one predecessor per operation. Some examples are presented in Sections 4.2 and 4.3. Constraints (3.17) and (3.17) operate similarly to Constraints (3.4) and (3.5), with the exception that they use DAG-modeled precedences rather than linearly modeled precedences.

It is also noteworthy to mention that Constrains (3.3) are no longer required because Constraints (3.17) and (3.18) are already accounting for precedence between operations of the same job, since the domain of $h$ includes $j$. Constraints (3.19), like Constraints (3.6) ensures that all operations assigned to machine $i$ that have no preceding operations, i.e. initial operations, begin only at a minimum time $Q_j$.

Similarly, Model-4 is a modification of Model-2, with OF as presented in (3.12) subject to (3.7)-(3.11) and (3.16)-(3.19), which also employs DAG-modeled precedences.

# Chapter 4

# Computational results

In this section, the performance of the proposed mathematical models is compared, across different approaches. Two sets of experiments were conducted.

The first set of experiments focuses on comparing the performance of the STF heuristic approach with the solutions obtained from Model-1 and Model-2 for different objective functions, such as Makespan and Deadline. By analyzing these results, insights can be gained into how effectively each mathematical model performs in terms of optimizing Makespan and Deadline. Comparing the solutions generated by Model-1 and Model-2 with the solutions given by STF allows for a comprehensive evaluation of their capabilities in meeting specific objectives, providing valuable information about the decision-making process. This comparison helps to identify the strengths and weaknesses of each approach, allowing investigations to be carried out to refine and improve upon them.

Based on the findings of the first set, the second set of experiments investigates how providing the solver with an initial heuristic solution (provided by STF) affects its solution development process (specifically the development of primal and dual solutions) and whether this approach can improve computational efficiency, solution quality or both. An expected outcome, e.g., would be a narrowing of the optimality gap, either by lowering the value of the OF (primal solution) or by increasing the best dual solution in a minimization problem. In addition, the second set introduces Model-3 and Model-4, which provide a much more flexible representation of the precedence between operations in the same job.

To conduct this comparison and present the findings in terms of objective function values, CPU times, and gap percentages, three types of datasets were used, namely Benchmark, Synthetic and Real-world based datasets, described respectively in Sections 4.1, 4.2 and 4.3. In the first set, three real-world based instances and a set of twenty benchmark modified datasets were used in the experiments. All instances used in the first set (instances using the suffix "_v1") were also used in the second set having their setup and processing times modified to allow for a wider range (instances with the suffix "_v2"), while keeping the same number of jobs, machines and operations, as well as the operations' precedence relationships, jobs start time constraints and deadlines.

The M parameter (also known as big M) was set for each instance to ten times the longer time between setup and processing times. As a result, it is not large enough to cause issues with precision and numerical representation in the programming languages used, but it is sufficient for the solver's behavior to be adequate and the constraints to be respected.

A time limit of three hours for first set and one hour for second set was imposed for the solver to find the optimal solution, returning the best viable solution found when this limit is reached.

The time required to the STF to generate a solution for each instance appears to be unaffected by their size. This independence arises from the utilization of simple heuristics for decision-making at each step, resulting in minimal temporal variations contingent upon the complexity of the given instance. Consequently, its temporal component was omitted in the second set of experiments, deeming it irrelevant (across all cases, the time involved consistently required less than one second) [23]. The emphasis is on prioritizing consideration of other factors that have a greater impact on the solution generation process.

The imposed time limit is an important practical consideration because, in real-world scenarios, the implications of scheduling decisions must be considered, and evaluating complex schedules with dozens of jobs and machines takes a long time. On top of that, it is often necessary to reschedule or make changes to an existing schedule, making it critical to generate a new production program so that the plant does not grind to a halt.

The experiments were run on AMD Ryzen 7 3700x 8-Core @3.60 GHz processor with 32GB RAM. To solve the models, Python language version 3.11.3 and Gurobi solver version 10.0.1 through the API provided by the gurobipy library were used. Experiments implementation details and all (benchmark, modified and generated) instances used are available at GitHub for first[1] and second[2] sets of the experiments.

Gurobi's default parameters were used, with no optimization or predefinition, with the exception of the parameter to focus on integrality, to ensure that solutions generated were integer.

[1] `https://github.com/thj3a/JobShopModels.py/tree/model-SBPO`
[2] `https://github.com/thj3a/JobShopModels.py`

## 4.1 Benchmark datasets

One of the main objectives of this work is to compare the effectiveness of the methods that we have developed for solving the FJSP. To do so, we need to use some datasets that can represent the characteristics and the constraints of the real cases that we are dealing with. However, finding such datasets is not an easy task (due to the companies' policy of not allowing their data to be publicly disclosed). So, most of the existing datasets in the literature are either too simple or too unrealistic.

Conversely, benchmark datasets are intentionally designed to be challenging for several reasons. These challenges serve to assess and compare the performance of algorithms, models, or techniques under realistic and demanding conditions. Overcoming these challenges demonstrates the robustness and effectiveness of a solution in addressing real-world complexities. Researchers and practitioners use benchmarking to understand the strengths and limitations of different approaches, fostering advancements in the field.

Twenty datasets that were derived from the work of FATTAHI *et al.* [31] were used. These datasets, however, lacked key information for this study, such as the SDST, job start times, and deadlines. As a result, we modified them by including the following information: To add the setup times, a random number from a normal distribution with a mean of zero and a standard deviation of ten was drawn and only the absolute number was used. To fulfill the start time constraint, similarly, an absolute random number was drawn from a normal distribution with a mean of ten and a standard deviation of two, with a probability of occurrence of 40%; otherwise, zero was used. The procedure used to include the deadlines, along with the modifications for the second set of experiments (instances with suffix "_v2") were the same described in Section 4.2.

Figures 4.1 and 4.2 demonstrate the precedence graphs of jobs 0 and 1, respectively, of the FSM_20_v2 instance, with their linear order of precedence.

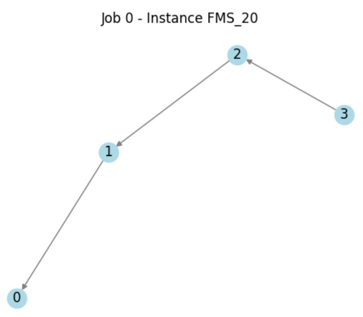

Figure 4.1: Graph representation of the precedence of the operations of instance FSM_20_v2's job 0.

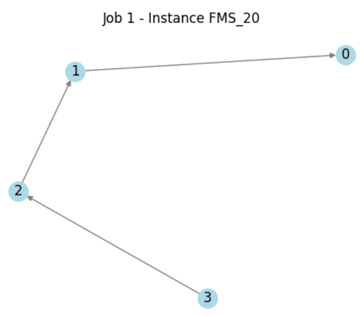

Figure 4.2: Graph representation of the precedence of the operations of instance FSM_20_v2's job 1.

The linear structure of precedence between operations sets apart this class from the others. This feature has not been modified from FATTAHI *et al.* [31]'s original instances and cannot represent the level of detail found in some real item structures.

## 4.2 Synthetic datasets

As shown in Section 2.3, the BA and URT graph types were employed to generate twenty datasets, each, explored in this work. These two types of graphs were particularly chosen because of their similar properties to real cases. While the properties of the URT graph can be observed in assembly plants, for example, the properties of the BA graph can be found in cases where the bill of materials (BOM) of the final products have few levels but are extremely complex.

The number of jobs, operations per job, and machines followed the distribution proposed in FATTAHI *et al.* [31] benchmark instances. The idea was to create instances with the same proportions, but with different characteristics in terms of the dependencies between the operations of each job. The precedence graphs were generated with the aid of the Networkx[3] library implemented in Python[4].

The number of alternative machines in a job that can process an operation was defined as the smallest value greater than one third of the total number of machines in the instance, while the machines that could process each operation were drawn from a uniform distribution among all the machines in the instance, without replacement.

The processing times were taken from a uniform distribution between two and five, while to generate synthetic setup times it was drawn a random number from a uniform distribution between the the minimum and maximum processing time

[3]`www.networkx.org`
[4]`www.python.org`

for each instance, considering all the jobs, operations and machines involved. Finally, the number drawn was multiplied by another random number, also uniformly distributed, between one and five. This resulted in a setup time that could be significantly different from the average processing time, depending on the instance characteristics and the random factors. This method has a lower expected value than just a uniform distribution between one and five times the higher processing time. The variance, on the other hand, is greater.

The earliest start time for each job and each operation was generated randomly from a uniform distribution between zero and three while the deadline for each job was calculated by multiplying the number of operations in the job by a random number drawn from a uniform distribution between the lower and higher processing times.

Figures 4.3, 4.4, depict the precedence graphs of jobs 9 and 11 from BA_20 instance, while 4.5 and 4.6 illustrate jobs 1 and 11 from URT_20 instance. The shape of the graphs representing precedence shows a clear difference. While the nodes in the BA_20 instance have a higher degree of entry, the graphs in the URT_-20 instance are, as expected, tree-shaped.

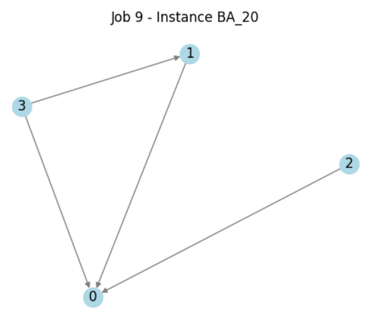

Figure 4.3: Graph representation of the precedence of the operations of instance BA_20's job 09.

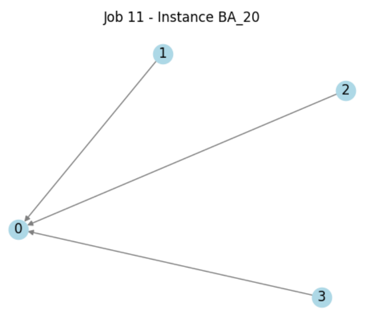

Figure 4.4: Graph representation of the precedence of the operations of instance BA_20's job 11.

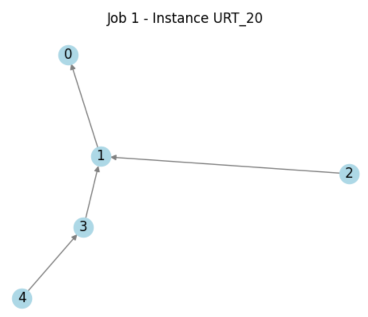

Figure 4.5: Graph representation of the precedence of the operations of instance URT_20's job 1.

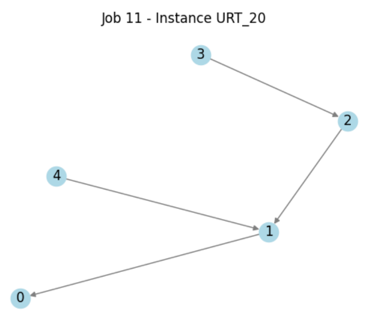

Figure 4.6: Graph representation of the precedence of the operations of instance URT_20's job 11.

## 4.3 Real-world based datasets

To create realistic and diverse instances of the problem, we used three datasets that were derived from real data collected from different industrial sectors. The first dataset came from the metalworking industry, which involves cutting, bending and assembling metal parts. The second dataset came from the plastic injection industry, which involves injecting molten plastic into molds to produce plastic parts. The third dataset came from the customized manufacturing industry, which involves producing unique products according to customer specifications. These three instances were simplified in terms of their constraints to fit the proposed models, while keeping their number of jobs, operations and machines, reflecting the different sizes, complexities and characteristics of the problem in different industrial contexts.

The number of operations per job, as shown in Figures 4.7 and 4.8, is one of the first characteristics that distinguishes graphs of real-world based instances from others. The graph for job 23 of the MetalMeca instance shows an interesting occurrence of operation precedence, with a bifurcation followed by a junction after a few operations.

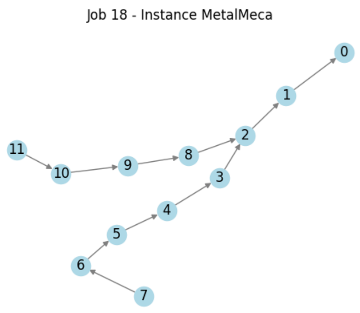

Figure 4.7: Graph representation of the precedence of the operations of instance MetalMeca_v2's job 18.

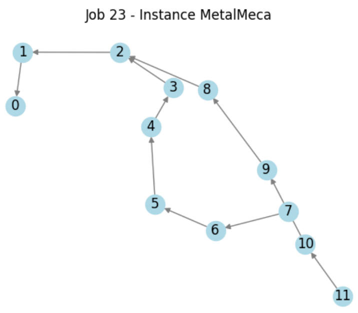

Figure 4.8: Graph representation of the precedence of the operations of instance MetalMeca_v2's job 23.

To use the data from these instances, a translation module from the STF data model to the data model used by the mathematical model had to be built. To accomplish this, time values were converted from seconds to minutes, and date values were translated based on the schedule's initial date (STF has a component that mimics a clock and controls the passage of time from a given date). Because STF contains other entities such as items, which can contain other attributes that do not belong to operations, such as quantity in stock, data relating to the precedence of operations also had to be remodeled.

## 4.4 Comparing STF with mathematical models

Table 4.1 shows characteristics of the instances and corresponding results. Important attributes of the instances are the total number of jobs ($n$), total number of operations ($\sum_j n_j$), and total number of machines ($m$). The density relating the number of binary variables (# Bin) with the total number of variables (# Var) is also shown. The objective function value (OF) and the time required by the solver is presented, in seconds (s), for the MILP and MINLP models and STF. The time mentioned does not include model construction in either the solver or the STF, as this time is irrelevant even for the largest instances (less than one second). Note that STF generates a single solution for each instance and does not depend on the objective function (thus, a single execution time). The optimality gap (Opt Gap) for the models is null whenever an optimal solution is found; otherwise, a non-negative gap means that the time limit of three hours was reached. The STF Heuristic Gap (Heur Gap) reflects the gap with respect to the OF value returned by the respective (Makespan or Deadline) MILP or MINLP model, being calculated by the mathematical formulas $\mathrm{OF}_{\mathrm{STF}} - \mathrm{OF}_{\mathrm{MILP}}/\mathrm{OF}_{\mathrm{MILP}}$ and $\mathrm{OF}_{\mathrm{STF}} - \mathrm{OF}_{\mathrm{MINLP}}/\mathrm{OF}_{\mathrm{MINLP}}$. Therefore, to compare the performance of MILP and MINLP models against STF, we should not compare the respective Gap columns directly, but column 6 with column 12 and column 9 with column 14, respectively.

Table 4.1: Computational results comparing mathematical models with STF solutions, without the use of DAGs, for both Makespan and Deadline OFs.

| | | | | | MILP and MINLP Models Solutions | | | | | | STF Heuristic Solutions | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instances | | | | | Makespan | | | Deadline | | | Makespan | | Deadline | | STF |
| Instance name | # Bin / # Var | $n$ | $\sum n_j$ | $m$ | OF (min) | Solver time (s) | Opt Gap | OF (min) | Solver time (s) | Opt Gap | OF (min) | Heur Gap | OF (min) | Heur Gap | Time (s) |
| FSM_01_v1 | 10 / 18 | 2 | 4 | 2 | 142 | 0.01 | 0.0% | 80 | 0.01 | 0.0% | 193 | 35.9% | 131 | 63.8% | 0.38 |
| FSM_02_v1 | 7 / 13 | 2 | 4 | 2 | 204 | 0.01 | 0.0% | 132 | 0.00 | 0.0% | 204 | 0.0% | 164 | 24.2% | 0.37 |
| FSM_03_v1 | 17 / 27 | 3 | 6 | 2 | 600 | 0.01 | 0.0% | 445 | 0.01 | 0.0% | 612 | 2.0% | 496 | 11.5% | 0.39 |
| FSM_04_v1 | 18 / 28 | 3 | 6 | 2 | 874 | 0.02 | 0.0% | 653 | 0.01 | 0.0% | 1001 | 14.5% | 878 | 34.5% | 0.38 |
| FSM_05_v1 | 22 / 34 | 3 | 6 | 2 | 303 | 0.04 | 0.0% | 221 | 0.03 | 0.0% | 321 | 5.9% | 298 | 34.8% | 0.38 |
| FSM_06_v1 | 96 / 112 | 3 | 9 | 3 | 808 | 0.02 | 0.0% | 626 | 0.03 | 0.0% | 1216 | 50.5% | 1085 | 73.3% | 0.39 |
| FSM_07_v1 | 99 / 118 | 3 | 9 | 5 | 923 | 0.01 | 0.0% | 696 | 0.01 | 0.0% | 952 | 3.1% | 921 | 32.3% | 0.43 |
| FSM_08_v1 | 99 / 118 | 3 | 9 | 4 | 655 | 0.04 | 0.0% | 519 | 0.06 | 0.0% | 773 | 18.0% | 813 | 56.6% | 0.41 |
| FSM_09_v1 | 99 / 118 | 3 | 9 | 3 | 553 | 0.07 | 0.0% | 443 | 0.07 | 0.0% | 579 | 4.7% | 566 | 27.8% | 0.38 |
| FSM_10_v1 | 38 / 58 | 4 | 12 | 5 | 1899 | 0.05 | 0.0% | 1599 | 0.04 | 0.0% | 2000 | 5.3% | 1883 | 17.8% | 0.40 |
| FSM_11_v1 | 81 / 114 | 5 | 15 | 6 | 2043 | 0.37 | 0.0% | 1703 | 0.41 | 0.0% | 2339 | 14.5% | 2402 | 41.0% | 0.43 |
| FSM_12_v1 | 102 / 141 | 5 | 15 | 7 | 1977 | 0.63 | 0.0% | 1620 | 0.61 | 0.0% | 2165 | 9.5% | 2333 | 44.0% | 0.40 |
| FSM_13_v1 | 149 / 197 | 6 | 18 | 7 | 2517 | 2.95 | 0.0% | 2078 | 8.85 | 0.0% | 3259 | 29.5% | 3474 | 67.2% | 0.41 |
| FSM_14_v1 | 191 / 247 | 7 | 21 | 7 | 3317 | 15.92 | 0.0% | 2741 | 45.86 | 0.0% | 3699 | 11.5% | 3939 | 43.7% | 0.42 |
| FSM_15_v1 | 196 / 251 | 7 | 21 | 7 | 3283 | 87.52 | 0.0% | 2714 | 36.63 | 0.0% | 3807 | 16.0% | 4062 | 49.7% | 0.41 |
| FSM_16_v1 | 244 / 306 | 8 | 24 | 7 | 4244 | 3469.86 | 0.0% | 3600 | 763.61 | 0.0% | 4949 | 16.6% | 5338 | 48.3% | 0.43 |
| FSM_17_v1 | 394 / 472 | 8 | 32 | 7 | 5953 | 10800.28 | 15.1% | 5290 | 10800.43 | 18.4% | 7268 | 22.1% | 7383 | 39.6% | 0.42 |
| FSM_18_v1 | 441 / 527 | 9 | 36 | 8 | 6720 | 10801.38 | 16.6% | 6088 | 10801.97 | 21.5% | 8237 | 22.6% | 8271 | 35.9% | 0.42 |
| FSM_19_v1 | 619 / 722 | 11 | 44 | 8 | 9709 | 10802.29 | 30.7% | 8785 | 10802.16 | 33.7% | 10911 | 12.4% | 10668 | 21.4% | 0.43 |
| FSM_20_v1 | 735 / 847 | 12 | 48 | 8 | 11597 | 10801.82 | 32.6% | 10256 | 10801.34 | 32.5% | 13005 | 12.1% | 12877 | 25.6% | 0.43 |
| FTQL_v1 | 47 / 71 | 5 | 24 | 5 | 70820 | 0.07 | 0.0% | 34533 | 0.12 | 0.0% | 88487 | 24.9% | 52200 | 51.2% | 0.43 |
| PlasticInject_v1 | 398 / 549 | 69 | 69 | 35 | 81298 | 101.66 | 0.0% | 70636 | 10801.38 | 1.4% | 89405 | 10.0% | 123417 | 74.7% | 0.57 |
| MetalMeca_v1 | 4791 / 5334 | 61 | 295 | 21 | 645591 | 10813.41 | 2.8% | 0 | 3.29 | 0.0% | 711671 | 10.2% | 440 | $***$ | 0.90 |

$***$ In this case, Heur Gap cannot be calculated, since there is no division by zero.

Table 4.1 also shows that optimal solutions were found for 16/20 (80%) benchmark instances and 2/3 (67%) real-world instances for both Model-1 and Model-2. For example, from instance FSM_17_v1 to FSM_20_v1 both the Model-1 and Model-2, the solver was not able to find the optimal solution to within the time limit provided. For these instances, the optimality gap provided ranges from 15% to 33%. Surprisingly, optimal or very small gap solutions were found for both the very large real-world instances MetalMeca_v1 and PlastInject_v1. It is also important to emphasize that for these instances, Heur Gaps were very large. In particular, this behavior occurred for almost all tested instances, i.e., Heur Gap was much larger than MILP Gap or MINLP Gap, for 22/23 (95,6%) and 23/23 (100%) of the tested instances, respectively. This indicates that scheduling under the solutions of the proposed models manages resources more efficiently, allowing for increased productive capacity without increasing the number working hours, for example. It is also worth noting that large real-world instances could be easier to solve by the MILP and MINLP models than the modified benchmark instances, due to actual constraints and parameters of real-world problems.

Certain instances within the benchmark class, on the other hand, have demonstrated an expected level of difficulty in their solution by MIP models, despite their relatively modest scale when compared to real-world counterparts. This observation highlights a crucial detail: the difficulty in solving benchmark problems extends beyond size considerations. In other words, the complexity and intricacies embedded within certain benchmark instances defy expectations, demonstrating that the degree of difficulty in problem-solving is not solely determined by the size of the problem at hand.

In contrast to the proposed model, and not surprising, STF found just a single optimal solution (for FSM_02_v1), although it provided good quality solutions (gap<10% in some instances). However, for most cases, Heuristic Gap is more than 10%, and up to more than 70% in the worst case, which is a brutal difference. This indicates that scheduling under the solutions of the proposed model manages resources more efficiently, allowing for increased productive capacity without increasing the number working hours, just by making the schedule more efficient.

It is worth noting that for some cases, besides meeting deadlines as an objective is more realistic and closer to reality in some shop-floor cases, the counterpart problem when minimizing makespan can be harder than the former (as can be seen in results for Model-1 and Model-2 for Instance MetalMeca_v1).

Figures 4.9 and 4.10 illustrate the schedule for the FSM_16_v1 instance generated by solving Model-1 and by STF, respectively. Despite the two schedules are similar, note that STF requires more time to finish all jobs and its gap is 16.6% with respect to the optimal solution found by the MILP model.

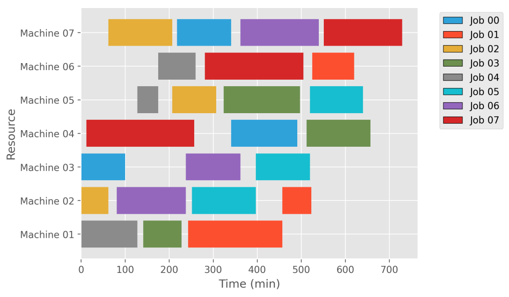

Figure 4.9: Model-1 Solution for instance FSM_16_v1.

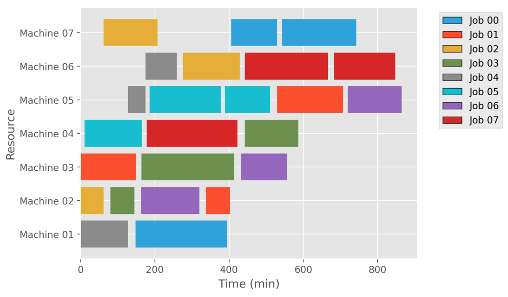

Figure 4.10: STF Heuristic Solution for instance FSM_16_v1.

Similarly, Figures 4.11 and 4.12 illustrate the schedule for the FSM_17_v1 instance generated by solving the MILP proposed model (makespan OF) and by STF. In contrast, note that the model required more time to finish all jobs (job 07 is the last one). However, by delaying the execution of job 07 the model finished many other jobs much earlier than STF, such as job 00 and job 06. Indeed, the heuristic gap considering the OF value is 22.1% with respect to the solution returned by the model. This kind of solution seems quite hard to obtain using heuristics.

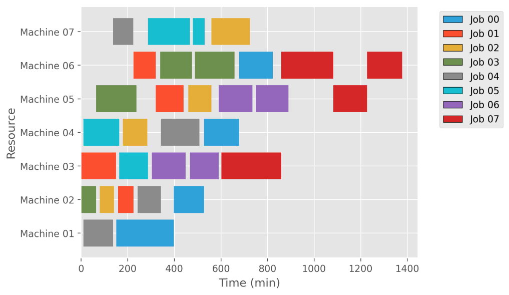

Figure 4.11: Model-1 Solution for instance FSM_17_v1.

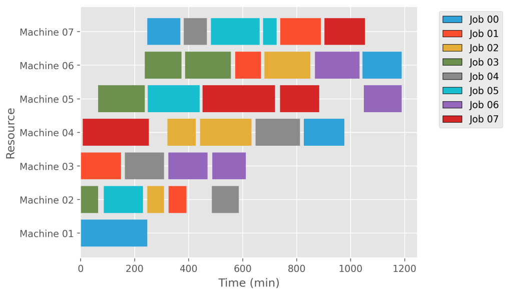

Figure 4.12: STF Heuristic Solution for instance FSM_17_v1.

Figures 4.13 and 4.14 illustrate the schedule for the FSM_16_v1 and FSM_-17_v1 instances generated by the Model-2. Note that for FSM_16_v1 the schedule generated is identical to the one generated by the MILP model (makespan OF), illustrated in Figure 4.9. In contrast, the schedule for FSM_17_v1 is much tighter than the corresponding schedule generated by the makespan OF (see Figure 4.11). Indeed, the deadline OF forces jobs to finish closer to their deadlines, as opposed to simply reducing their finish times.

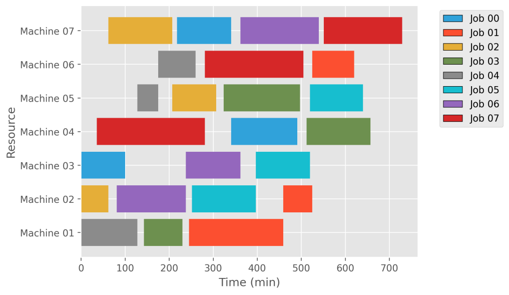

Figure 4.13: Model-2 Solution for instance FSM_16_v1.

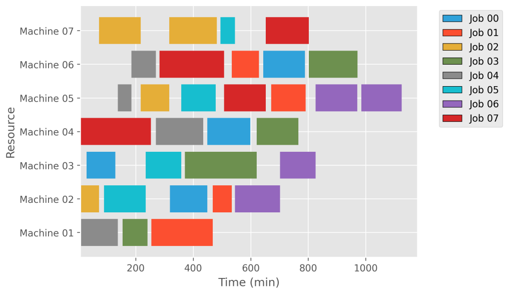

Figure 4.14: Model-2 Solution for instance FSM_17_v1.

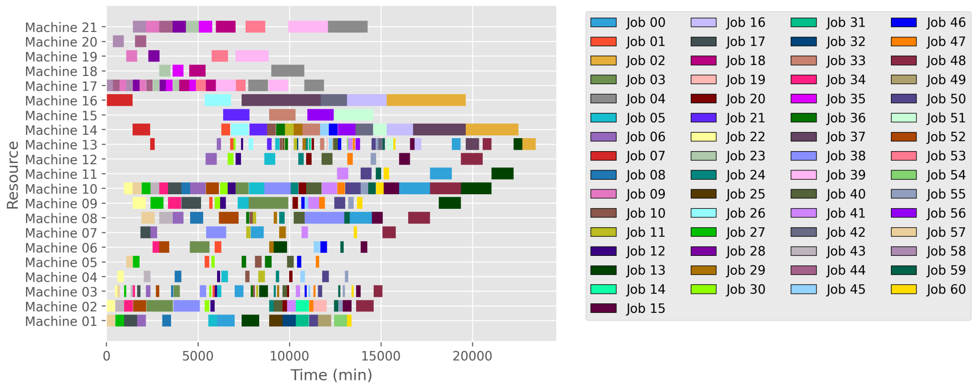

Figure 4.15: Model-1 Solution for instance MetalMeca_v1.

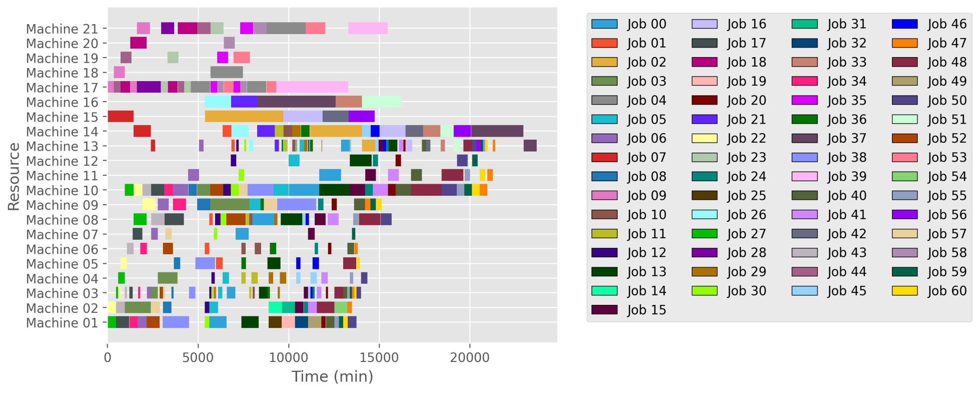

Figure 4.16: STF Heuristic Solution for instance MetalMeca_v1.

Figures 4.15 and 4.16 illustrate the schedule for the MetalMeca_v1 instance (with 61 jobs) generated by solving the proposed MILP model (makespan OF) and by STF, respectively. While the two schedules look relatively similar and the heuristic gap is 10.2%, note that the schedule generated by the model has fewer jobs finishing after time 15000 in comparison to STF, where many jobs finish after the same threshold. Again, this shows the potential of using a mathematical model and a solver to generate efficient schedules, even when the obtained solution is not optimal, as in this scenario.

The distinctions between real and benchmark instances become clear when the figures 4.10 and 4.16 are assessed. Despite the similarities between FSM_16_v1 and FSM_17_v1, the small distinction of a few additional operations per job is what appears to make instance FSM_17_v1 considerably more difficult to handle in the proposed amount of time, an expected behavior when dealing with MILP and MINLP models. However, when comparing instances with large differences in the number of machines, jobs, and operations, such as FSM_17_v1 and MetalMeca_-v1, it is clear that the variances in processing and setup times make it easier to find solutions closer to the global optimum for larger real-world based instances.

An important practical consideration is the amount of time required to generate a solution. As Table 4.1 shows, while in all scenarios where an optimal solution was found, less than 100 seconds was required (with the exception of FSM_16_v1 and PlasticInject_v1) to solve the respective model, in the cases where the optimal solution was not found, the model ran until the three hour time limit. Lastly, for the benchmark instances, as the instance size increases, the difficulty to solve it also increases.

In contrast, Table 4.1 also shows that the execution time required by STF to generate a solution was less than one second in all problem instances. Moreover, its execution time does not seem to have (strong) correlation with the problem size. This is clearly a large advantage of this approach, that quickly yields a feasible solution using a greedy procedure driven by multiple heuristics. However, its execution time depends more on the number of the competing operations and the number of heuristics and their potential combinations (a limited number of heuristics was used here) than the total number of jobs, operations and machines. Lastly, it is important to emphasize that the quality of these solutions generated by STF was, for the vast majority of instances, much lower than the quality of the solutions generated by the mathematical models.

The time required to build a solution using STF heuristics is its most significant advantage. With an substantial increase in the size of the instances, there is virtually no temporal fluctuation. However, as stated in Section 1.2, multiple heuristics can be employed, and the search for the set of heuristics can take a long time depending on the number of heuristics available and their potential combinations.

Finally, understanding the problem and how to properly apply the heuristic approach are important aspects in evaluating how long it will take a user to construct a reasonably good solution in a reasonable amount of time, since the solutions provided by STF kernel may appear to be satisfactory, there is no way to prove how far it is from its global optimum without the aid of a mathematical model.

## 4.5 Combining STF with mathematical models

The results of combining the STF with the MIP models Model-3 and Model-4 introduced in Section 3.3 are shown in this section.

Similarly to Table 4.1, Table 4.2 presents for each instance, the ratio of binary variables (# Bin) to total model variables(# Var) and its characteristics with the difference that only results for the makespan objective are displayed. The following table also compares the results for Model-3 with the use and the absence of STF solution together with the characteristics of all instances tested in this set of experiments. There are three additional columns containing the difference of objection function (Diff OF), time (Diff Time) and gap (Diff Gap) between the results obtained with (STF+MIP) and without (MIP) the STF solution. It is crucial to emphasize that the time numbers on both tables only represent the amount of time the solver takes; they do not account for the amount of time STF takes to produce the solution that is supplied to the solver. Negative values indicate that using the STF+MIP strategy produced better results than using the MIP approach alone.

Table 4.2: Computational results for Model-3 comparing the use of the STF solution.

| Instances | | | | | MIP | | | STF+MIP | | | Comparison | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance name | # Bin / # Var | $n$ | $m$ | $\sum n_j$ | OF (min) | Solver Time (s) | Opt. Gap | OF (min) | Solver Time (s) | Opt. Gap | Diff OF | Diff Time | Diff Gap |
| BA_01 | 20 / 25 | 2 | 2 | 4 | 44 | 0.00 | 0.00% | 44 | 0.00 | 0.00% | 0 | 0.00 | 0.00% |
| BA_02 | 20 / 25 | 2 | 2 | 4 | 48 | 0.00 | 0.00% | 48 | 0.00 | 0.00% | 0 | 0.00 | 0.00% |
| BA_03 | 42 / 49 | 3 | 2 | 6 | 72 | 0.00 | 0.00% | 72 | 0.01 | 0.00% | 0 | 0.00 | 0.00% |
| BA_04 | 42 / 49 | 3 | 2 | 6 | 73 | 0.00 | 0.00% | 73 | 0.00 | 0.00% | 0 | 0.00 | 0.00% |
| BA_05 | 42 / 49 | 3 | 2 | 6 | 101 | 0.00 | 0.00% | 101 | 0.01 | 0.00% | 0 | 0.00 | 0.00% |
| BA_06 | 90 / 100 | 3 | 3 | 9 | 100 | 0.01 | 0.00% | 100 | 0.01 | 0.00% | 0 | 0.00 | 0.00% |
| BA_07 | 99 / 118 | 3 | 5 | 9 | 45 | 0.14 | 0.00% | 45 | 0.14 | 0.00% | 0 | 0.01 | 0.00% |
| BA_08 | 99 / 118 | 3 | 4 | 9 | 46 | 0.32 | 0.00% | 46 | 0.32 | 0.00% | 0 | 0.00 | 0.00% |
| BA_09 | 90 / 100 | 3 | 3 | 9 | 98 | 0.01 | 0.00% | 98 | 0.01 | 0.00% | 0 | 0.00 | 0.00% |
| BA_10 | 168 / 193 | 4 | 5 | 12 | 65 | 0.50 | 0.00% | 65 | 0.71 | 0.00% | 0 | 0.21 | 0.00% |
| BA_11 | 255 / 286 | 5 | 6 | 15 | 81 | 1.13 | 0.00% | 81 | 1.06 | 0.00% | 0 | -0.07 | 0.00% |
| BA_12 | 270 / 316 | 5 | 7 | 15 | 64 | 4.63 | 0.00% | 64 | 4.18 | 0.00% | 0 | -0.45 | 0.00% |
| BA_13 | 378 / 433 | 6 | 7 | 18 | 86 | 175.91 | 0.00% | 86 | 773.99 | 0.00% | 0 | 598.08 | 0.00% |
| BA_14 | 504 / 568 | 7 | 7 | 21 | 121 | 3600.11 | 19.83% | 121 | 3600.14 | 19.83% | 0 | 0.03 | 0.00% |
| BA_15 | 504 / 568 | 7 | 7 | 21 | 112 | 3600.23 | 22.32% | 112 | 3600.24 | 22.32% | 0 | 0.02 | 0.00% |
| BA_16 | 648 / 721 | 8 | 7 | 24 | 146 | 3600.32 | 45.21% | 146 | 3600.22 | 45.21% | 0 | -0.10 | 0.00% |
| BA_17 | 1120 / 1217 | 8 | 7 | 32 | 218 | 3600.46 | 61.01% | 206 | 3600.32 | 58.25% | -12 | -0.13 | -2.76% |
| BA_18 | 1404 / 1513 | 9 | 8 | 36 | 231 | 3600.51 | 56.71% | 231 | 3600.47 | 56.71% | 0 | -0.04 | 0.00% |
| BA_19 | 2068 / 2201 | 11 | 8 | 44 | 325 | 3600.56 | 68.31% | 325 | 3600.78 | 68.31% | 0 | 0.22 | 0.00% |
| BA_20 | 2448 / 2593 | 12 | 8 | 48 | 381 | 3600.74 | 67.45% | 397 | 3600.69 | 69.02% | 16 | -0.05 | 1.56% |
| URT_01 | 20 / 25 | 2 | 2 | 4 | 44 | 0.02 | 0.00% | 44 | 0.00 | 0.00% | 0 | -0.02 | 0.00% |
| URT_02 | 20 / 25 | 2 | 2 | 4 | 42 | 0.02 | 0.00% | 42 | 0.02 | 0.00% | 0 | 0.00 | 0.00% |
| URT_03 | 42 / 49 | 3 | 2 | 6 | 69 | 0.04 | 0.00% | 69 | 0.04 | 0.00% | 0 | -0.01 | 0.00% |
| URT_04 | 42 / 49 | 3 | 2 | 6 | 77 | 0.03 | 0.00% | 77 | 0.04 | 0.00% | 0 | 0.01 | 0.00% |
| URT_05 | 42 / 49 | 3 | 2 | 6 | 101 | 0.06 | 0.00% | 101 | 0.05 | 0.00% | 0 | -0.01 | 0.00% |
| URT_06 | 90 / 100 | 3 | 3 | 9 | 97 | 0.03 | 0.00% | 97 | 0.03 | 0.00% | 0 | 0.00 | 0.00% |
| URT_07 | 99 / 118 | 3 | 5 | 9 | 43 | 0.76 | 0.00% | 43 | 0.76 | 0.00% | 0 | -0.01 | 0.00% |
| URT_08 | 99 / 118 | 3 | 4 | 9 | 48 | 3600.19 | 12.50% | 48 | 3600.43 | 12.50% | 0 | 0.25 | 0.00% |
| URT_09 | 90 / 100 | 3 | 3 | 9 | 136 | 0.07 | 0.00% | 136 | 0.07 | 0.00% | 0 | -0.01 | 0.00% |
| URT_10 | 168 / 193 | 4 | 5 | 12 | 66 | 2.36 | 0.00% | 66 | 2.21 | 0.00% | 0 | -0.15 | 0.00% |
| URT_11 | 255 / 286 | 5 | 6 | 15 | 93 | 4.40 | 0.00% | 93 | 4.14 | 0.00% | 0 | -0.26 | 0.00% |
| URT_12 | 270 / 316 | 5 | 7 | 15 | 72 | 33.92 | 0.00% | 72 | 31.98 | 0.00% | 0 | -1.95 | 0.00% |
| URT_13 | 378 / 433 | 6 | 7 | 18 | 92 | 736.05 | 0.00% | 92 | 717.88 | 0.00% | 0 | -18.16 | 0.00% |
| URT_14 | 504 / 568 | 7 | 7 | 21 | 126 | 3600.20 | 20.63% | 126 | 3600.18 | 20.63% | 0 | -0.01 | 0.00% |
| URT_15 | 504 / 568 | 7 | 7 | 21 | 120 | 3600.08 | 15.83% | 120 | 3600.31 | 15.83% | 0 | 0.22 | 0.00% |
| URT_16 | 648 / 721 | 8 | 7 | 24 | 150 | 3600.16 | 37.33% | 152 | 3600.21 | 38.82% | 2 | 0.05 | 1.48% |
| URT_17 | 1120 / 1217 | 8 | 7 | 32 | 222 | 3600.44 | 52.70% | 225 | 3600.38 | 55.56% | 3 | -0.06 | 2.85% |
| URT_18 | 1404 / 1513 | 9 | 8 | 36 | 237 | 3600.44 | 54.01% | 238 | 3600.99 | 53.78% | 1 | 0.54 | -0.23% |
| URT_19 | 2068 / 2201 | 11 | 8 | 44 | 328 | 3600.91 | 61.59% | 311 | 3600.60 | 59.49% | -17 | -0.32 | -2.10% |
| URT_20 | 3756 / 3913 | 12 | 9 | 60 | 6081 | 3600.26 | 0.13% | 6081 | 3600.79 | 0.10% | 0 | 0.52 | -0.03% |
| FSM_01_v2 | 24 / 33 | 2 | 2 | 4 | 1231 | 0.13 | 0.00% | 1231 | 0.12 | 0.00% | 0 | -0.01 | 0.00% |
| FSM_02_v2 | 22 / 29 | 2 | 2 | 4 | 1339 | 0.05 | 0.00% | 1339 | 0.05 | 0.00% | 0 | 0.01 | 0.00% |
| FSM_03_v2 | 46 / 57 | 3 | 2 | 6 | 2620 | 0.11 | 0.00% | 2620 | 0.11 | 0.00% | 0 | 0.00 | 0.00% |
| FSM_04_v2 | 46 / 57 | 3 | 2 | 6 | 3152 | 0.15 | 0.00% | 3152 | 0.15 | 0.00% | 0 | 0.00 | 0.00% |
| FSM_05_v2 | 48 / 61 | 3 | 2 | 6 | 2027 | 0.26 | 0.00% | 2027 | 0.27 | 0.00% | 0 | 0.02 | 0.00% |
| FSM_06_v2 | 96 / 112 | 3 | 3 | 9 | 3306 | 3600.33 | 2.90% | 3306 | 3600.10 | 2.90% | 0 | -0.22 | 0.00% |
| FSM_07_v2 | 99 / 118 | 3 | 5 | 9 | 2932 | 0.40 | 0.00% | 2932 | 0.52 | 0.00% | 0 | 0.12 | 0.00% |
| FSM_08_v2 | 99 / 118 | 3 | 4 | 9 | 2365 | 0.44 | 0.00% | 2365 | 3600.13 | 3.72% | 0 | 3599.68 | 3.72% |
| FSM_09_v2 | 99 / 118 | 3 | 3 | 9 | 2290 | 0.69 | 0.00% | 2290 | 0.70 | 0.00% | 0 | 0.02 | 0.00% |
| FSM_10_v2 | 164 / 185 | 4 | 5 | 12 | 4834 | 0.29 | 0.00% | 4834 | 1.39 | 0.00% | 0 | 1.10 | 0.00% |
| FSM_11_v2 | 258 / 292 | 5 | 6 | 15 | 5602 | 5.99 | 0.00% | 5602 | 6.72 | 0.00% | 0 | 0.73 | 0.00% |

Continued on next page

Table 4.2 – Continued from previous page

| Instances | | | | | MIP | | | STF+MIP | | | Comparison | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance name | # Bin / # Var | $n$ | $m$ | $\sum n_j$ | OF (min) | Solver Time (s) | Opt. Gap | OF (min) | Solver Time (s) | Opt. Gap | Diff OF | Diff Time | Diff Gap |
| FSM_12_v2 | 264 / 304 | 5 | 7 | 15 | 5308 | 4.80 | 0.00% | 5308 | 11.38 | 0.00% | 0 | 6.57 | 0.00% |
| FSM_13_v2 | 372 / 421 | 6 | 7 | 18 | 7247 | 72.91 | 0.00% | 7247 | 66.13 | 0.00% | 0 | -6.78 | 0.00% |
| FSM_14_v2 | 497 / 554 | 7 | 7 | 21 | 8581 | 190.20 | 0.00% | 8581 | 148.45 | 0.00% | 0 | **-41.75** | 0.00% |
| FSM_15_v2 | 496 / 552 | 7 | 7 | 21 | 8885 | 141.11 | 0.00% | 8885 | 158.20 | 0.00% | 0 | 17.09 | 0.00% |
| FSM_16_v2 | 638 / 701 | 8 | 7 | 24 | 12045 | 3600.27 | 12.98% | 12007 | 3600.15 | 16.07% | -38 | -0.12 | 3.10% |
| FSM_17_v2 | 1102 / 1181 | 8 | 7 | 32 | 16465 | 3600.39 | 36.59% | 15504 | 3600.05 | 35.53% | **-961** | -0.34 | -1.05% |
| FSM_18_v2 | 1382 / 1469 | 9 | 8 | 36 | 16896 | 3600.07 | 32.32% | 16786 | 3600.08 | 32.07% | **-110** | 0.01 | -0.25% |
| FSM_19_v2 | 2039 / 2143 | 11 | 8 | 44 | 26260 | 3600.38 | 48.86% | 25833 | 3600.88 | 47.15% | **-427** | 0.51 | -1.70% |
| FSM_20_v2 | 2416 / 2529 | 12 | 8 | 48 | 29677 | 3600.30 | 48.37% | 31837 | 3600.29 | 52.54% | 2160 | -0.01 | 4.17% |
| FTQL_v2 | 600 / 625 | 5 | 5 | 24 | 74452 | 0.84 | 0.00% | 74452 | 0.70 | 0.00% | 0 | -0.14 | 0.00% |
| PlasticInject_v2 | 6758 / 6956 | 69 | 35 | 81 | 2934826 | 3600.83 | 9.20% | 2930911 | 3600.71 | 9.20% | **-3915** | -0.13 | 0.00% |
| MetalMeca_v2 | 87568 / 88112 | 41 | 21 | 295 | 511907 | 3600.41 | 11.84% | 515707 | 3600.44 | 12.38% | 3800 | 0.03 | 0.54% |

Similarly, Table 4.3 shows the results for Model-4 (deadline OF).

Table 4.3: Computational results for Model-4 comparing the use of the STF solution.

| Instances | | | | | MIP | | | STF+MIP | | | Comparison | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance name | # Bin / # Var | $n$ | $m$ | $\sum n_j$ | OF (min) | Solver Time (s) | Opt. Gap | OF (min) | Solver Time (s) | Opt. Gap | Diff OF | Diff Time | Diff Gap |
| BA_01 | 22 / 27 | 2 | 2 | 4 | 28 | 0.00 | 0.00% | 28 | 0.00 | 0.00% | 0.00 | 0.00 | 0.00% |
| BA_02 | 22 / 27 | 2 | 2 | 4 | 30 | 0.00 | 0.00% | 30 | 0.01 | 0.00% | 0.00 | 0.00 | 0.00% |
| BA_03 | 45 / 52 | 3 | 2 | 6 | 60 | 0.00 | 0.00% | 60 | 0.00 | 0.00% | 0.00 | 0.00 | 0.00% |
| BA_04 | 45 / 52 | 3 | 2 | 6 | 53 | 0.00 | 0.00% | 53 | 0.00 | 0.00% | 0.00 | 0.00 | 0.00% |
| BA_05 | 45 / 52 | 3 | 2 | 6 | 75 | 0.01 | 0.00% | 75 | 0.01 | 0.00% | 0.00 | 0.00 | 0.00% |
| BA_06 | 93 / 103 | 3 | 3 | 9 | 73 | 0.00 | 0.00% | 73 | 0.02 | 0.00% | 0.00 | 0.01 | 0.00% |
| BA_07 | 105 / 124 | 3 | 5 | 9 | 11 | 0.17 | 0.00% | 11 | 0.06 | 0.00% | 0.00 | -0.10 | 0.00% |
| BA_08 | 105 / 124 | 3 | 4 | 9 | 25 | 0.36 | 0.00% | 25 | 0.15 | 0.00% | 0.00 | -0.21 | 0.00% |
| BA_09 | 93 / 103 | 3 | 3 | 9 | 74 | 0.01 | 0.00% | 92 | 0.01 | 0.00% | 18.00 | 0.00 | 0.00% |
| BA_10 | 176 / 201 | 4 | 5 | 12 | 29 | 0.72 | 0.00% | 29 | 0.45 | 0.00% | 0.00 | -0.28 | 0.00% |
| BA_11 | 265 / 296 | 5 | 6 | 15 | 34 | 0.90 | 0.00% | 34 | 0.69 | 0.00% | 0.00 | -0.21 | 0.00% |
| BA_12 | 285 / 331 | 5 | 7 | 15 | 20 | 4.85 | 0.00% | 20 | 4.98 | 0.00% | 0.00 | 0.13 | 0.00% |
| BA_13 | 396 / 451 | 6 | 7 | 18 | 19 | 44.32 | 0.00% | 19 | 22.33 | 0.00% | 0.00 | -21.99 | 0.00% |
| BA_14 | 525 / 589 | 7 | 7 | 21 | 63 | 3600.13 | 61.90% | 63 | 3600.04 | 58.73% | 0.00 | -0.09 | -3.17% |
| BA_15 | 525 / 589 | 7 | 7 | 21 | 41 | 3600.03 | 4.88% | 41 | 3600.03 | 12.20% | 0.00 | -0.01 | 7.32% |
| BA_16 | 672 / 745 | 8 | 7 | 24 | 58 | 3600.45 | 94.83% | 50 | 3600.04 | 78.00% | -8.00 | -0.40 | -16.83% |
| BA_17 | 1144 / 1241 | 8 | 7 | 32 | 98 | 3600.25 | 95.92% | 98 | 3600.22 | 95.92% | 0.00 | -0.03 | 0.00% |
| BA_18 | 1431 / 1540 | 9 | 8 | 36 | 133 | 3600.16 | 93.23% | 133 | 3600.07 | 93.23% | 0.00 | -0.09 | 0.00% |
| BA_19 | 2101 / 2234 | 11 | 8 | 44 | 175 | 3600.24 | 98.86% | 175 | 3600.18 | 98.86% | 0.00 | -0.06 | 0.00% |
| BA_20 | 2484 / 2629 | 12 | 8 | 48 | 249 | 3600.28 | 97.99% | 249 | 3600.40 | 97.99% | 0.00 | 0.12 | 0.00% |
| URT_01 | 22 / 27 | 2 | 2 | 4 | 28 | 0.02 | 0.00% | 28 | 0.00 | 0.00% | 0.00 | -0.02 | 0.00% |
| URT_02 | 22 / 27 | 2 | 2 | 4 | 32 | 0.02 | 0.00% | 32 | 0.02 | 0.00% | 0.00 | 0.00 | 0.00% |
| URT_03 | 45 / 52 | 3 | 2 | 6 | 47 | 0.06 | 0.00% | 47 | 0.06 | 0.00% | 0.00 | 0.00 | 0.00% |
| URT_04 | 45 / 52 | 3 | 2 | 6 | 51 | 0.00 | 0.00% | 51 | 0.00 | 0.00% | 0.00 | 0.00 | 0.00% |
| URT_05 | 45 / 52 | 3 | 2 | 6 | 75 | 0.08 | 0.00% | 75 | 0.06 | 0.00% | 0.00 | -0.02 | 0.00% |
| URT_06 | 93 / 103 | 3 | 3 | 9 | 70 | 0.04 | 0.00% | 70 | 0.06 | 0.00% | 0.00 | 0.01 | 0.00% |

Continued on next page

Table 4.3 – Continued from previous page

| Instances | | | | | MIP | | | STF+MIP | | | Comparison | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance name | # Bin / # Var | $n$ | $m$ | $\sum n_j$ | OF (min) | Solver Time (s) | Opt. Gap | OF (min) | Solver Time (s) | Opt. Gap | Diff OF | Diff Time | Diff Gap |
| URT_07 | 105 / 124 | 3 | 5 | 9 | 10 | 0.33 | 0.00% | 10 | 0.33 | 0.00% | 0.00 | 0.00 | 0.00% |
| URT_08 | 105 / 124 | 3 | 4 | 9 | 29 | 0.33 | 0.00% | 29 | 0.37 | 0.00% | 0.00 | 0.04 | 0.00% |
| URT_09 | 93 / 103 | 3 | 3 | 9 | 112 | 0.06 | 0.00% | 112 | 0.05 | 0.00% | 0.00 | 0.00 | 0.00% |
| URT_10 | 176 / 201 | 4 | 5 | 12 | 33 | 0.86 | 0.00% | 33 | 0.84 | 0.00% | 0.00 | -0.01 | 0.00% |
| URT_11 | 265 / 296 | 5 | 6 | 15 | 44 | 3.84 | 0.00% | 44 | 3.72 | 0.00% | 0.00 | -0.13 | 0.00% |
| URT_12 | 285 / 331 | 5 | 7 | 15 | 28 | 12.12 | 0.00% | 28 | 11.90 | 0.00% | 0.00 | -0.22 | 0.00% |
| URT_13 | 396 / 451 | 6 | 7 | 18 | 22 | 28.76 | 0.00% | 22 | 96.33 | 0.00% | 0.00 | 67.57 | 0.00% |
| URT_14 | 525 / 589 | 7 | 7 | 21 | 58 | 3600.08 | 46.55% | 58 | 3600.03 | 51.72% | 0.00 | -0.05 | 5.17% |
| URT_15 | 525 / 589 | 7 | 7 | 21 | 53 | 3600.06 | 50.94% | 53 | 1328.84 | 0.00% | 0.00 | **-2271.21** | **-50.94%** |
| URT_16 | 672 / 745 | 8 | 7 | 24 | 58 | 3600.05 | 81.03% | 58 | 3600.05 | 81.03% | 0.00 | 0.00 | 0.00% |
| URT_17 | 1144 / 1241 | 8 | 7 | 32 | 102 | 3600.58 | 94.12% | 116 | 3600.08 | 93.10% | 14.00 | -0.50 | -1.01% |
| URT_18 | 1431 / 1540 | 9 | 8 | 36 | 143 | 3600.11 | 86.01% | 145 | 3600.12 | 91.03% | 2.00 | 0.01 | 5.02% |
| URT_19 | 2101 / 2234 | 11 | 8 | 44 | 204 | 3600.10 | 96.08% | 204 | 3600.33 | 96.08% | 0.00 | 0.23 | 0.00% |
| URT_20 | 3768 / 3925 | 12 | 9 | 60 | 5928 | 3600.30 | 0.13% | 5926 | 3600.78 | 0.20% | -2.00 | 0.48 | 0.07% |
| FSM_01_v2 | 28 / 37 | 2 | 2 | 4 | 271 | 0.08 | 0.00% | 271 | 0.07 | 0.00% | 0.00 | -0.01 | 0.00% |
| FSM_02_v2 | 25 / 32 | 2 | 2 | 4 | 379 | 0.03 | 0.00% | 379 | 0.02 | 0.00% | 0.00 | -0.01 | 0.00% |
| FSM_03_v2 | 51 / 62 | 3 | 2 | 6 | 1180 | 0.14 | 0.00% | 1180 | 0.15 | 0.00% | 0.00 | 0.01 | 0.00% |
| FSM_04_v2 | 51 / 62 | 3 | 2 | 6 | 1712 | 0.10 | 0.00% | 1712 | 0.08 | 0.00% | 0.00 | -0.03 | 0.00% |
| FSM_05_v2 | 54 / 67 | 3 | 2 | 6 | 587 | 0.35 | 0.00% | 587 | 0.35 | 0.00% | 0.00 | 0.00 | 0.00% |
| FSM_06_v2 | 101 / 117 | 3 | 3 | 9 | 1866 | 0.22 | 0.00% | 1866 | 0.21 | 0.00% | 0.00 | 0.00 | 0.00% |
| FSM_07_v2 | 105 / 124 | 3 | 5 | 9 | 1492 | 0.29 | 0.00% | 1492 | 0.37 | 0.00% | 0.00 | 0.08 | 0.00% |
| FSM_08_v2 | 105 / 124 | 3 | 4 | 9 | 925 | 0.32 | 0.00% | 925 | 0.32 | 0.00% | 0.00 | -0.01 | 0.00% |
| FSM_09_v2 | 105 / 124 | 3 | 3 | 9 | 850 | 0.31 | 0.00% | 850 | 3600.71 | 1.29% | 0.00 | 3600.41 | 1.29% |
| FSM_10_v2 | 171 / 192 | 4 | 5 | 12 | 2914 | 3600.33 | 2.57% | 2914 | 3600.55 | 2.57% | 0.00 | 0.22 | 0.00% |
| FSM_11_v2 | 269 / 303 | 5 | 6 | 15 | 3202 | 1.84 | 0.00% | 3202 | 4.64 | 0.00% | 0.00 | 2.80 | 0.00% |
| FSM_12_v2 | 277 / 317 | 5 | 7 | 15 | 2908 | 16.83 | 0.00% | 2908 | 6.68 | 0.00% | 0.00 | -10.15 | 0.00% |
| FSM_13_v2 | 388 / 437 | 6 | 7 | 18 | 4367 | 703.75 | 0.00% | 4367 | 179.15 | 0.00% | 0.00 | **-524.60** | 0.00% |
| FSM_14_v2 | 516 / 573 | 7 | 7 | 21 | 5221 | 343.41 | 0.00% | 5221 | 1468.89 | 0.00% | 0.00 | 1125.47 | 0.00% |
| FSM_15_v2 | 516 / 572 | 7 | 7 | 21 | 5525 | 374.61 | 0.00% | 5525 | 1000.99 | 0.00% | 0.00 | 626.38 | 0.00% |
| FSM_16_v2 | 661 / 724 | 8 | 7 | 24 | 8170 | 3600.43 | 30.61% | 8167 | 3600.16 | 34.37% | -3.00 | -0.27 | 3.76% |
| FSM_17_v2 | 1118 / 1197 | 8 | 7 | 32 | 11644 | 3600.29 | 42.52% | 11644 | 3600.15 | 42.37% | 0.00 | -0.14 | -0.15% |
| FSM_18_v2 | 1400 / 1487 | 9 | 8 | 36 | 13941 | 3601.08 | 51.55% | 13656 | 3600.40 | 49.50% | **-285.00** | -0.68 | **-2.05%** |
| FSM_19_v2 | 2061 / 2165 | 11 | 8 | 44 | 21838 | 3600.46 | 63.01% | 19495 | 3600.53 | 56.36% | **-2343.00** | 0.07 | **-6.66%** |
| FSM_20_v2 | 2440 / 2553 | 12 | 8 | 48 | 26230 | 3600.44 | 67.09% | 25604 | 3600.61 | 65.89% | **-626.00** | 0.17 | **-1.20%** |
| FTQL_v2 | 605 / 630 | 5 | 5 | 24 | 47572 | 0.86 | 0.00% | 47572 | 0.88 | 0.00% | 0.00 | 0.02 | 0.00% |
| PlasticInject_v2 | 6908 / 7106 | 69 | 35 | 81 | 2936384 | 3600.24 | 10.34% | 2936384 | 3600.83 | 10.33% | 0.00 | 0.59 | -0.01% |
| MetalMeca_v2 | 87609 / 88153 | 41 | 21 | 295 | 58147 | 3600.31 | 84.83% | 61574 | 3600.37 | 84.19% | 3427.00 | 0.06 | -0.64% |

As a first observation, it can be seen that the benchmark and real-world instances tested on the second set of experiments had worse results overall than in the first set of experiments, in terms of optimal gaps. While the time limit set in this set was only a third of the one in the first set, no correlation can exist since the instances themselves are not the same. So, we can only note that the number of variables has significantly increased, but as previously stated, this does not indicate that the difficulty of solving the model has increased.

For benchmark instances, the greatest improvements when solving Model-3 were seen on FSM_16_v2 to FSM_19_v2, with 38, 961, 110, and 427 minutes saved in scheduling, respectively. This means that the sum of the total time required to finish all production tasks was decreased by this value, translating to a significant amount of time saved (more than 16 hours in the best case). Additionally, the reduced scheduling time on these benchmark instances indicates that the model is becoming more efficient and effective in solving complex problems. These time savings can be allocated to the increase the amount of items produced or to perform scheduled maintenance on machines. Rather, the optimality gaps persisted at high values for these cases, suggesting that the problem's dual solution or the quality of the primal solution can still be enhanced.

Likewise, for instances FSM_18_v2 to FSM_20_v2 when solving Model-4, a huge reduction was observed in OF value. Instance FSM_19, specially, owns the highest reduction of OF value of all instances tested for the Deadline OF, a reduction of more than 10% in previous value. It means that the factory's overall punctuality was improved, even though more jobs were completed later, i.e., the solution better exploited time windows for items that could be delivered later, reducing the overall tardiness.

Figures 4.17 and 4.18 show that the best solution has a schedule much tighter in the first half of scheduling, even with longer time windows between tasks occurring more frequently.

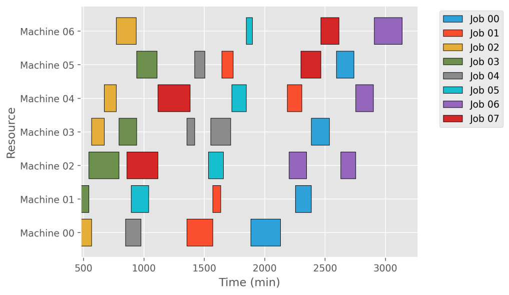

Figure 4.17: Gantt graph with the scheduling for Instance FSM_17_v2 for Model-3 without STF's initial solution.

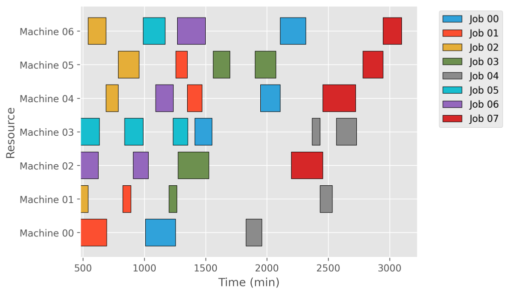

Figure 4.18: Gantt graph with the scheduling for Instance FSM_17_v2 for Model-3 with STF's initial solution.

The best solution, however, is clearly tighter in the comparison shown in Figures 4.19 and 4.20. Though, the difference in the time windows and overall tightness between tasks when comparing FSM_17_v2 and FSM_19_v2 with FSM_17_v1 and FSM_17_v1 is very clear.

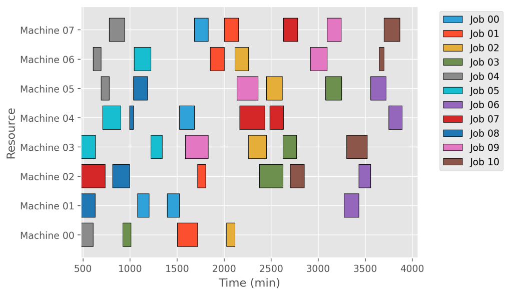

Figure 4.19: Gantt graph with the scheduling for Instance FSM_19_v2 for Model-4 without STF's initial solution.

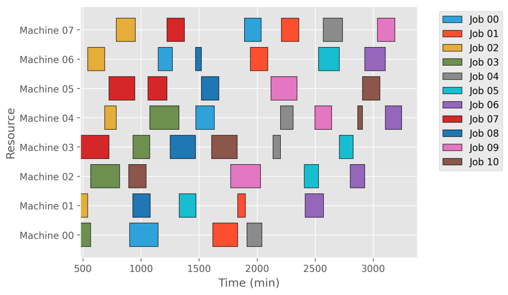

Figure 4.20: Gantt graph with the scheduling for Instance FSM_19_v2 for Model-4 with STF's initial solution.

With regard to synthetic instances, the significant improvement showcased by solving Model-4 at Instance URT_15 highlights the importance of utilizing effective heuristic solutions like those provided by STF combined with MIP approaches. Figures 4.21 and 4.22 show that the solutions in this case are the same. Figures 4.23 and 4.24, however, illustrate that, although the primal solution reached its optimal value later (1025s over 205s) than in the case where no initial solution was provided, the development of the dual solution improved much faster over time. In the worst case, the best dual solution did not reach half of its optimal value, even though it took three times longer than the best case. With the aid of the STF solution, the solver was able to prove more quickly that the solution found without its help was already the optimal one, saving approximately 37 minutes of processing time and reducing the optimality gap in incredibly 51%.

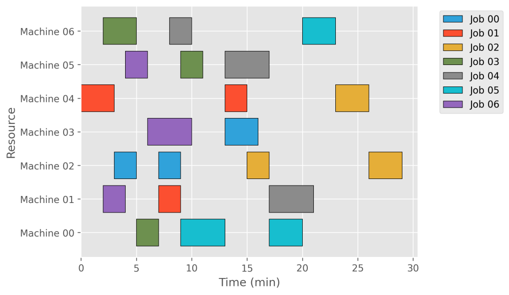

Figure 4.21: Gantt graph with the scheduling for Instance URT_15 for Model-4 without STF's initial solution.

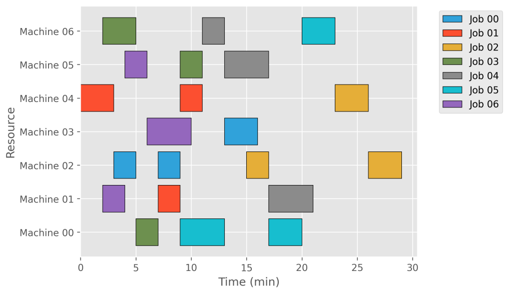

Figure 4.22: Gantt graph with the scheduling for Instance URT_15 for Model-4 with STF's initial solution.

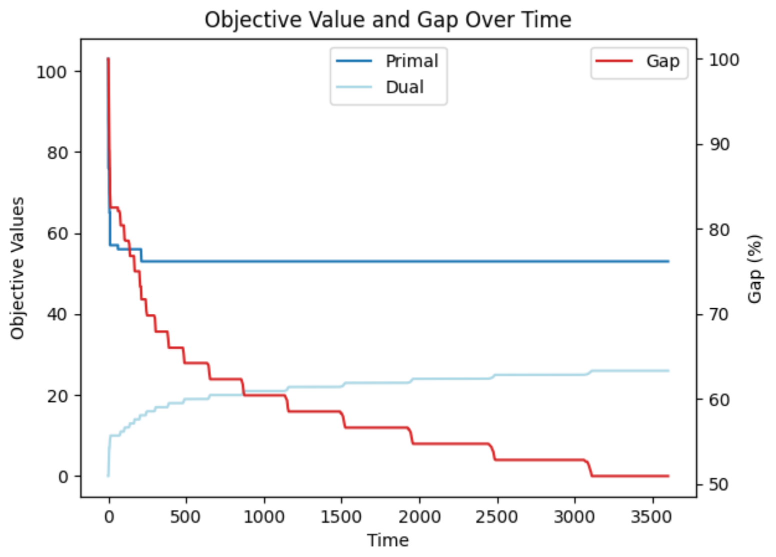

Figure 4.23: Evolution of the primal and dual solutions of Model-4 for the Instance URT_15 without the aid of STF's solution.

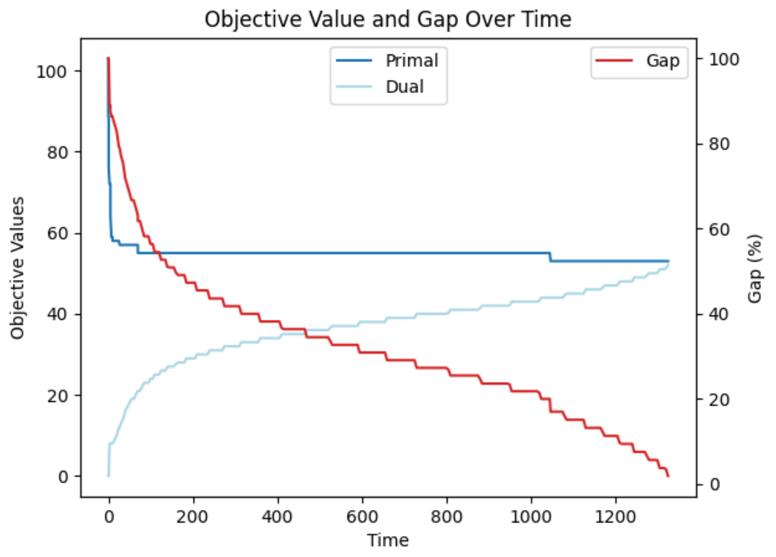

Figure 4.24: Evolution of the primal and dual solutions of Model-4 for the Instance URT_15 with the aid of STF's solution.

Another successful case was on the BA_16 instance, where the optimality gap was reduced by 16%. Indeed Figures 4.26 shows a tighter Gantt graph that 4.25. It's worth noting that the optimality gaps, particularly when solving Model-4, were unexpectedly large for the largest synthetic instances, reaching close to 99%. Although these values are extremely high, this does not imply that the solutions found are of low quality; rather, it indicates that the distance between the dual solution and the primal solution found is still substantial. However, as in the case of the

URT_15 instance, if better initial solutions are provided or more time is given to the solver, it can be proven that the solution found is optimal.

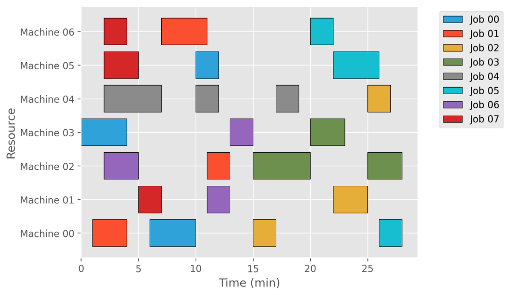

Figure 4.25: Gantt graph with the scheduling for Instance BA_16 for Model-4 without the aid of STF's solution.

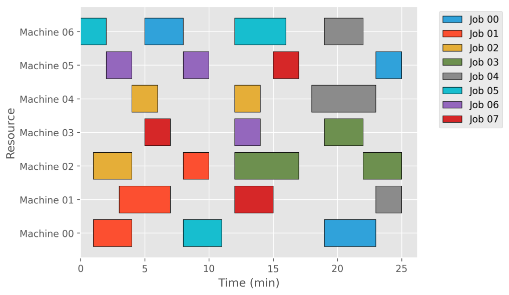

Figure 4.26: Gantt graph with the scheduling for Instance BA_16 for Model-4 with the aid of STF's solution.

Figures 4.27 and 4.28 show that when the size of the instances is very large, the process of visually understanding solutions becomes extremely complex. When it comes to real-world examples, no noteworthy changes have occurred in terms of the gap and time required differences. Despite that, what appears to be a modest reduction of only approximately 0.13% in the OF value for Model-3 of PlasticInjection_v2 instance, results in more than 60 hours of free resources for various operations. This aspect is specially relevant in long term planning scenarios, where the time limit is larger, but also the solution sought is the best possible, since the impacts caused by this type of decision are deeper and more extensive.

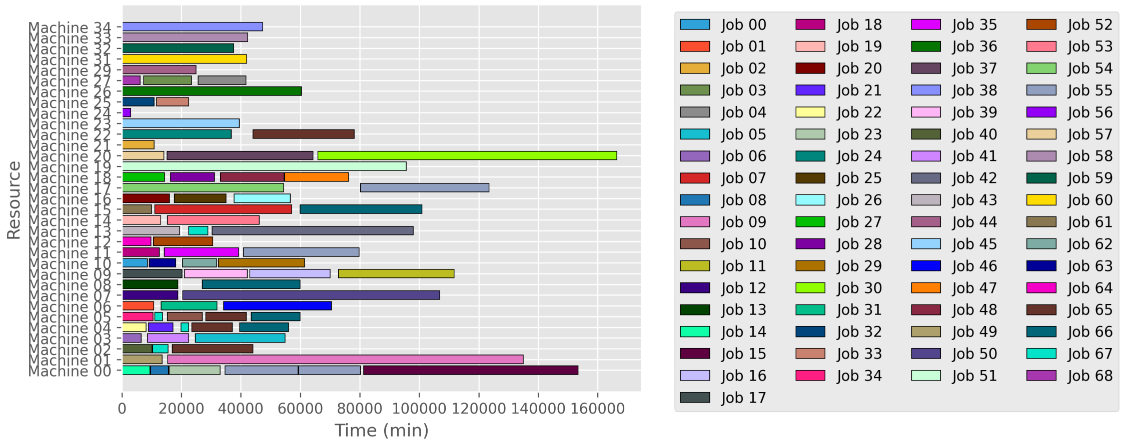

Figure 4.27: Gantt graph with the scheduling for Instance PlasticInjection_v2 for Model-3 without the aid of STF's solution.

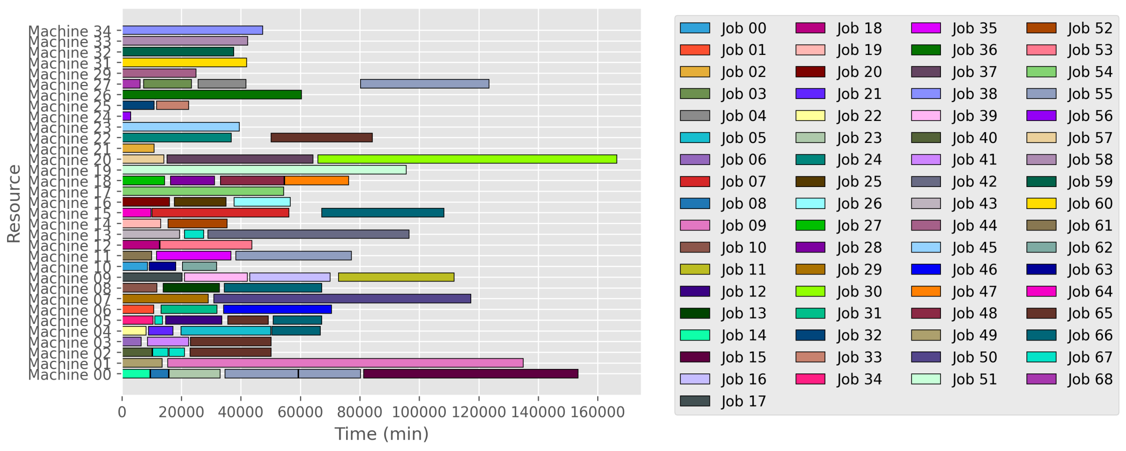
Figure 4.28: Gantt graph with the scheduling for Instance PlasticInjection_v2 for Model-3 with the aid of STF's solution.

Nonetheless, an curious phenomenon occurs as illustrated in Figures 4.29 and 4.30: the STF solution continues to have a positive impact on how the solver handles the branch-and-bound process in the pursuit of the optimal solution even after several minutes. Because of the early cut of some nodes, the STF solution becomes even more valuable in such cases, reducing the intricate maze of possibilities the solver needs to explore, resulting in a more unstable evolution, but also allowing for greater improvements on optimality gap. Figure 4.29 depicts imperceptible gains in primal and dual solutions unless the optimality gap is examined. This effect is more visible in Figure 4.30, around the 2400s. This is particularly crucial when dealing with complex optimization problems where deterministic methods may struggle, and significant advancements on primal and dual solutions occur scarcely.

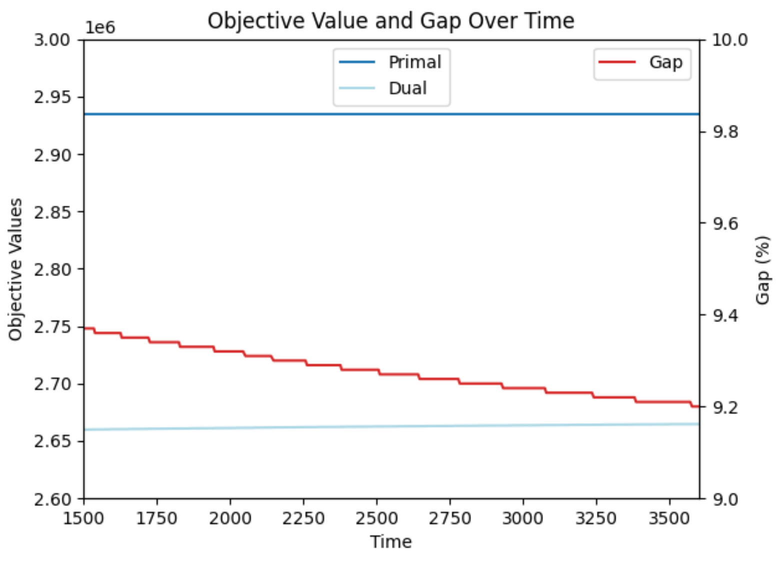

Figure 4.29: Evolution of the primal and dual solutions of Model-4 for the Instance PlasticInject_v2 without the aid of STF's solution.

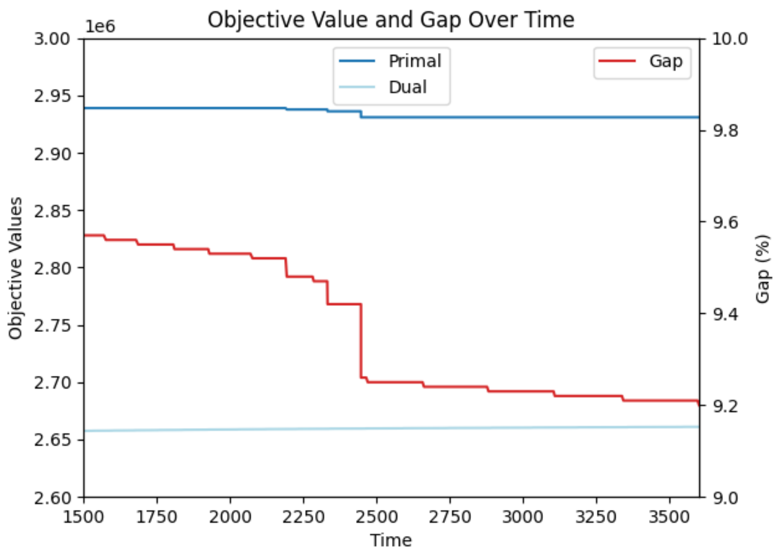

Figure 4.30: Evolution of the primal and dual solutions of Model-4 for the Instance PlasticInject_v2 with the aid of STF's solution.

Despite the fact that applying the STF solution as the first primal solution to the problem had a number of beneficial effects, there were some situations where the best viable solution found had drawbacks. The evolution of the primal and dual solutions for Instance URT_20 when solving Model-3 is depicted in Figures 4.31 and 4.32. This could be a result of limited exploration space. Reducing the space can occasionally have a positive effect, but it can also take longer to arrive at better solutions. It is evident that even though the dual solution has not yet reached optimality, the primal solution is optimal; as a result, the gap is still non-zero even though it is very small. Solving Model-3 for the MetalMeca_v2 instance is another instance of an even more harmful event. In this case, we observe that even though the optimality gap is smaller than in the case where the initial solution is not given, the value of the primal solution when the initial solution was employed ends up being higher, resulting in a worse feasible solution.

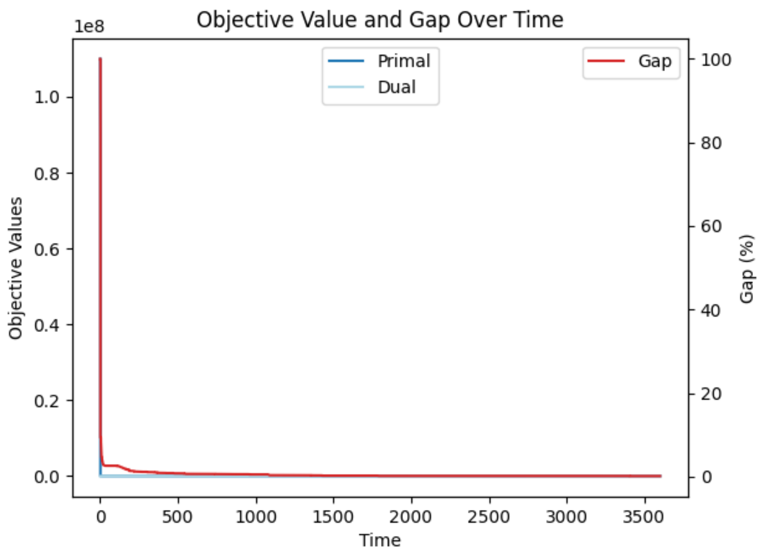

Figure 4.31: Evolution of the primal and dual solutions of Model-3 for the URT_20 instance without the aid of STF's solution.

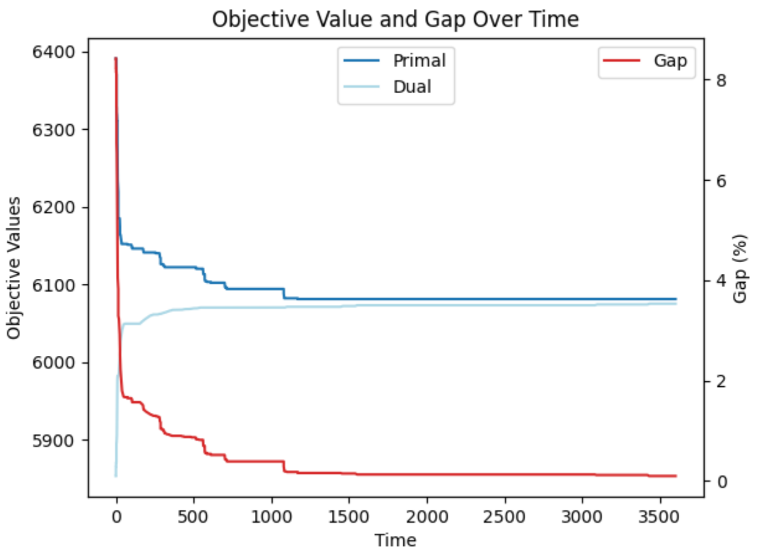

Figure 4.32: Evolution of the primal and dual solutions of Model-3 for the instance URT_20 with the aid of STF's solution.

# Chapter 5

# Conclusions and future work

The Flexible Job Shop Scheduling Problem (FJSP) is a fundamental problem in modern manufacturing since solutions can provide efficient plans and schedules for assembly and production lines, reducing resource consumption and working hours. An important consideration in FJSP is accurately representing real world scenarios which are often very complex and specific. The complexity is increased when complex real-world constraints like minimum start times, different objective functions and complex precedence relations are faced.

This paper presents an in-depth study of this problem and proposes four mathematical models to tackle these challenges with realistic constraints such as minimum start and setup times, different objective functions such as deadlines, and complex precedence relations among operations.

Models 1 and 3 are based on mixed integer linear programming, whereas Models 2 and 4 are based on Mixed Integer Programming due to the presence of non-linear components in their formulation. Models 3 and 4 are also based on directed acyclic graphs. A mathematical solver is used to solve all proposed models using the branch-and-bound method.

Model-1 is a MILP model that aims to minimize the FJSP's makespan. It is based on assigning operations of a job to a machine, the order of precedence between operations of different jobs on the same machine, and the time each operation must begin.

Model-2 adds a new decision variable to Model-1 to measure the tardiness of each job, assuming that each job has a due date.

Model-3 and Model-4 are variations of Model-1 and Model-2, respectively, with the addition of a new parameter to assign a precedence constraint between operations of the same job, rather than assuming a linear one, which adds a new level of complexity to the problem.

The performance of the proposed Model-1 and Model-2 is also compared to that of state-of-the-art software that employs a heuristic core STF based on discrete event simulation.

The paper evaluates the two first models on benchmark and real-world based instances in a first set of experiments and shows that the proposed models can obtain optimal or near-optimal solutions for medium-sized problems and outperform heuristic solutions in all tested cases.

Numerical evaluations suggest that primary and dual solutions can be beneficially affected by reducing their relative distances through the use of better initial heuristics than those provided by the solver.

In the second set of experiments, the paper compares the last two models on benchmark, synthetic, and real-world instances, with the addition of a given solution provided by STF. The integration with the discrete event based simulation core and has demonstrated that MIP models results can be significantly improved, though unexpected effects can still occur, i.e., the use of the STF as an initial solution provider did not always result in improvement; in those cases, initial solutions based on the solver's own heuristics are likely to produce better results.

This work also demonstrates that the proposed models can capture the key characteristics and challenges of real-world manufacturing environments and provide effective and efficient solutions to the FJSP. As a result, the improved performance of the models demonstrates its improved ability to handle larger and more sophisticated scheduling tasks, being capable of handling very complex operations precedence structures and specific shop floor characteristics. This progress is promising for future applications in various scenarios where time and resource management are crucial.

## 5.1 Future work

While the models proposed in this work embody important constraints, other aspects of complex and real production lines should also be incorporated, thus allowing the models to be used in a broader context. Therefore, a suggestion for future work is the incorporation of features such as representing possible idle times after an operation (e.g., drying before folding), representing transportation time between two operations of a job in different machines, and allowing for flexible batch processing (e.g., produce 100 items in two operations of 50 items) are also requirements often needed in manufacturing scenarios.

One of the method's limitations is that, while STF solutions can be useful in many cases, there is still room for improvement in the solutions that are provided to the solver. As a result, techniques such as metaheuristics could be used to improve the initial solutions presented to the solver. Furthermore, the solver's settings could be optimized, such as allocating more time to its initial heuristics or focusing on improving the dual solution.

More effective comparisons, such as directly comparing models with the same OFs but different precedence representations, could be made. In addition, the STF heuristic solution could be passed to Model-1 and Model-2 and the results observed. Instead of STF heuristics, metaheuristics could be implemented in the STF kernel to provide higher-quality solutions.

The complexity of scheduling is not solely technical but also involves dealing with human factors. Involving the workforce and managing their skills, shift preferences, and workload is crucial to creating a schedule that is both efficient and practical. Employee morale, motivation, and work-life balance can all be impacted by the scheduling decisions, making workforce management an integral part of the scheduling process.

The use of preemption by mathematical models could be investigated further in order to gain insights into how to apply it on heuristics and metaheuristics that can provide better initial solutions and speed up the resolution of complex and large instances.

Further research is required to assess the difference in complexity between models with and without the use of DAGs under the same time constraint. Finally, multi-objective functions could be explored to tackle both OFs covered in this work, as well as others.

# References

[1] ZHENG, T., ARDOLINO, M., BACCHETTI, A., et al. “The applications of Industry 4.0 technologies in manufacturing context: a systematic literature review”, *International Journal of Production Research*, v. 59, n. 6, pp. 1922–1954, 2021. doi: 10.1080/00207543.2020.1824085. Disponível em: <`https://doi.org/10.1080/00207543.2020.1824085`>.

[2] WANG, Y., MA, H.-S., YANG, J.-H., et al. “Industry 4.0: a way from mass customization to mass personalization production”, *Advances in manufacturing*, v. 5, pp. 311–320, 2017. doi: https://doi.org/10.1007/s40436-017-0204-7. Disponível em: <`https://link.springer.com/article/10.1007/s40436-017-0204-7#citeas`>.

[3] DUGUAY, C. R., LANDRY, S., PASIN, F. “From mass production to flexible/agile production”, *International Journal of Operations & Production Management*, v. 17, n. 12, pp. 1183–1195, 1997.

[4] SLACK, N. “The changing nature of operations flexibility”, *International Journal of Operations & Production Management*, v. 25, n. 12, pp. 1201–1210, 2005.

[5] JOVANOVIC, B., GILBERT, R. J. “The diversification of production”, *Brookings papers on economic activity. Microeconomics*, v. 1993, n. 1, pp. 197–247, 1993.

[6] LENSTRA, J., RINNOOY KAN, A., BRUCKER, P. “Complexity of Machine Scheduling Problems”. In: Hammer, P., Johnson, E., Korte, B., et al. (Eds.), *Studies in Integer Programming*, v. 1, *Annals of Discrete Mathematics*, Elsevier, pp. 343–362, 1977. doi: https://doi.org/10.1016/S0167-5060(08)70743-X. Disponível em: <`https://www.sciencedirect.com/science/article/pii/S016750600870743X`>.

[7] JIANG, Z., YUAN, S., MA, J., et al. “The evolution of production scheduling from Industry 3.0 through Industry 4.0”, *International Journal of Production Research*, v. 60, n. 11, pp. 3534–3554, 2022.

[8] CARVALHO, A. N., SCAVARDA, L. F., LUSTOSA, L. J. "Implementing finite capacity production scheduling: lessons from a practical case", *International Journal of Production Research*, v. 52, n. 4, pp. 1215–1230, 2014.

[9] PARENTE, M., FIGUEIRA, G., AMORIM, P., et al. "Production scheduling in the context of Industry 4.0: review and trends", *International Journal of Production Research*, v. 58, n. 17, pp. 5401–5431, 2020.

[10] XIE, J., GAO, L., PENG, K., et al. "Review on flexible job shop scheduling", *IET Collaborative Intelligent Manufacturing*, v. 1, n. 3, pp. 67–77, 2019.

[11] MENG, L., ZHANG, C., SHAO, X., et al. "MILP models for energy-aware flexible job shop scheduling problem", *Journal of cleaner production*, v. 210, pp. 710–723, 2019.

[12] BIRGIN, E. G., FEOFILOFF, P., FERNANDES, C. G., et al. "A MILP model for an extended version of the flexible job shop problem", *Optimization Letters*, v. 8, pp. 1417–1431, 2014.

[13] HAM, A. M., CAKICI, E. "Flexible job shop scheduling problem with parallel batch processing machines: MIP and CP approaches", *Computers & Industrial Engineering*, v. 102, pp. 160–165, 2016.

[14] REN, H., TANG, L. "An improved hybrid MILP/CP algorithm framework for the job-shop scheduling". In: *2009 IEEE international conference on automation and logistics*, pp. 890–894. IEEE, 2009.

[15] DEMIR, Y., İŞLEYEN, S. K. "Evaluation of mathematical models for flexible job-shop scheduling problems", *Applied Mathematical Modelling*, v. 37, n. 3, pp. 977–988, 2013.

[16] JONES, A., RABELO, L. C., SHARAWI, A. T. "Survey of job shop scheduling techniques", *NISTIR, National Institute of Standards and Technology, Gaithersburg, MD*, 1998.

[17] SHEN, X.-N., YAO, X. "Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems", *Information Sciences*, v. 298, pp. 198–224, 2015.

[18] VITAL SOTO, A. V. *Flexible job-shop scheduling problem with sequencing flexibility: Mathematical models and solution algorithms*. Tese de Doutorado, University of Windsor, 2019.

[19] ZHANG, S., WANG, S. “Flexible Assembly Job-Shop Scheduling With Sequence-Dependent Setup Times and Part Sharing in a Dynamic Environment: Constraint Programming Model, Mixed-Integer Programming Model, and Dispatching Rules”, *IEEE Transactions on Engineering Management*, v. 65, n. 3, pp. 487–504, 2018. doi: 10.1109/TEM.2017.2785774.

[20] MENG, L., ZHANG, C., REN, Y., et al. “Mixed-integer linear programming and constraint programming formulations for solving distributed flexible job shop scheduling problem”, *Computers  Industrial Engineering*, v. 142, pp. 106347, 2020. ISSN: 0360-8352. doi: https://doi.org/10.1016/j.cie.2020.106347. Disponível em: <`https://www.sciencedirect.com/science/article/pii/S0360835220300814`>.

[21] TÜRKYILMAZ, A., ŞENVAR, Ö., ÜNAL, İ., et al. “A research survey: heuristic approaches for solving multi objective flexible job shop problems”, *Journal of Intelligent Manufacturing*, v. 31, pp. 1949–1983, 2020.

[22] GOMES, M., BARBOSA-PÓVOA, A., NOVAIS, A. “Optimal scheduling for flexible job shop operation”, *International Journal of Production Research*, v. 43, n. 11, pp. 2323–2353, 2005. doi: 10.1080/00207540412331330101. Disponível em: <`https://doi.org/10.1080/00207540412331330101`>.

[23] SANTANA, A., FIGUEIREDO, D., BAHIENSE, L. “MATHEMATICAL MODELING FOR THE JOB SHOP SCHEDULING PROBLEM WITH REALISTIC CONSTRAINTS”. In: *LV Simpósio Brasileiro de Pesquisa Operacional (SBPO)*, 2023.

[24] PINEDO, M. L. *Scheduling*. Stern School of Business, Dept. Information, Operations , New York University, New York, USA, Springer International Publishing, 2016. Disponível em: <`http://link.springer.com/10.1007/978-3-319-26580-3`>.

[25] PINHA, D. C., AHLUWALIA, R. S., CARVALHO, A. N., et al. “Supply Chain Scheduling: A motorcycle assembly case study”, *IFAC-PapersOnLine*, v. 48, n. 3, pp. 1527–1532, 2015.

[26] CHAUDHRY, I. A., KHAN, A. A. “A research survey: review of flexible job shop scheduling techniques”, *International Transactions in Operational Research*, v. 23, n. 3, pp. 551–591, 2016. Disponível em: <`https://onlinelibrary.wiley.com/doi/abs/10.1111/itor.12199`>.

[27] SHEN, L., DAUZÈRE-PÉRÈS, S., NEUFELD, J. S. “Solving the flexible job shop scheduling problem with sequence-dependent setup times”, *Euro-*

*pean Journal of Operational Research*, v. 265, pp. 503–516, mar. 2018. doi: 10.1016/j.ejor.2017.08.021. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S037722171730752X>.

[28] KIM, S., BOBROWSKI, P. "Impact of sequence-dependent setup time on job shop scheduling performance", *The International Journal of Production Research*, v. 32, n. 7, pp. 1503–1520, 1994.

[29] ROSHANAEI, V., AZAB, A., ELMARAGHY, H. "Mathematical modelling and a meta-heuristic for flexible job shop scheduling", *International Journal of Production Research*, v. 51, n. 20, pp. 6247–6274, out. 2013. doi: 10.1080/00207543.2013.827806. Disponível em: <http://www.tandfonline.com/doi/abs/10.1080/00207543.2013.827806>.

[30] YEUNG, W.-K., CHOI, T.-M., CHENG, T. C. E. "Optimal Scheduling of a Single-Supplier Single-Manufacturer Supply Chain With Common due Windows", *IEEE Transactions on Automatic Control*, v. 55, pp. 2767–2777, 2010. doi: 10.1109/TAC.2010.2049766. Disponível em: <http://ieeexplore.ieee.org/document/5460987/>.

[31] FATTAHI, P., SAIDI MEHRABAD, M., JOLAI, F. "Mathematical modeling and heuristic approaches to flexible job shop scheduling problems", *Journal of intelligent manufacturing*, v. 18, pp. 331–342, 2007.

[32] WINKLEHNER, P., HAUDER, V. A. "Flexible job-shop scheduling with release dates, deadlines and sequence dependent setup times: a real-world case", *Procedia Computer Science*, v. 200, pp. 1654–1663, 2022. doi: 10.1016/j.procs.2022.01.366. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1877050922003751>.

[33] XIULI, W., JUNJIAN, P., ZIRUN, X., et al. "An improved multi-objective optimization algorithm for solving flexible job shop scheduling problem with variable batches", *Journal of Systems Engineering and Electronics*, v. 32, pp. 272–285, 2021. doi: 10.23919/JSEE.2021.000024. Disponível em: <https://ieeexplore.ieee.org/document/9430105/>.

[34] ÖZGÜVEN, C., ÖZBAKIR, L., YAVUZ, Y. "Mathematical models for job-shop scheduling problems with routing and process plan flexibility", *Applied Mathematical Modelling*, v. 34, n. 6, pp. 1539–1548, 2010.

[35] JUNG, S., WOO, Y.-B., KIM, B. S. "Two-stage assembly scheduling problem for processing products with dynamic component-sizes and a setup time", *Computers & Industrial Engineering*, v. 104, pp. 98–113, 2017.

doi: 10.1016/j.cie.2016.12.030. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0360835216305083>.

[36] VITAL-SOTO, A., AZAB, A., BAKI, M. F. "Mathematical modeling and a hybridized bacterial foraging optimization algorithm for the flexible job-shop scheduling problem with sequencing flexibility", *Journal of Manufacturing Systems*, v. 54, pp. 74–93, 2020.

[37] TUTUMLU, B., SARAÇ, T. "A MIP model and a hybrid genetic algorithm for flexible job-shop scheduling problem with job-splitting", *Computers Operations Research*, v. 155, pp. 106222, 2023. doi: https://doi.org/10.1016/j.cor.2023.106222. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054823000862>.

[38] SCHLENKRICH, M., PARRAGH, S. N. "Solving large scale industrial production scheduling problems with complex constraints: an overview of the state-of-the-art", *Procedia Computer Science*, v. 217, pp. 1028–1037, 2023.

[39] VERDERAME, P. M., ELIA, J. A., LI, J., et al. "Planning and scheduling under uncertainty: a review across multiple sectors", *Industrial & engineering chemistry research*, v. 49, n. 9, pp. 3993–4017, 2010.

[40] CAO, Z., LIN, C., ZHOU, M. "A Knowledge-Based Cuckoo Search Algorithm to Schedule a Flexible Job Shop with Sequencing Flexibility", *IEEE Transactions on Automation Science and Engineering*, v. 18, n. 1, pp. 56 – 69, 2021. doi: 10.1109/TASE.2019.2945717.

[41] ALBERT, R., BARABÁSI, A.-L. "Statistical mechanics of complex networks", *Reviews of modern physics*, v. 74, n. 1, pp. 47, 2002.

[42] ZADOROZHNYI, V. N., YUDIN, E. B. "Structural properties of the scale-free Barabasi-Albert graph", *Automation and Remote Control*, v. 73, pp. 702–716, 2012.