Ana Cristina Lima Gomes

Matheuristics applied to multi-objective production scheduling in a steel industry

Belo Horizonte, Brazil

March 2021

Ana Cristina Lima Gomes

Matheuristics applied to multi-objective production scheduling in a steel industry

Thesis presented to the Graduate Program in Electrical Engineering of the Universidade Federal de Minas Gerais in fulfillment of the requirements for the degree of Doctor in Electrical Engineering.

Universidade Federal de Minas Gerais

Department of Electrical Engineering

Graduate Program in Electrical Engineering

Supervisor: Prof. Dr. Eduardo Gontijo Carrano

Co-supervisor: Prof. Dr. Martín Gómez Ravetti

Belo Horizonte, Brazil

March 2021

G633m

Gomes, Ana Cristina Lima.
Matheuristics applied to multi-objective product scheduling in a steel industry [recurso eletrônico] / Ana Cristina Lima Gomes. - 2021.
1 recurso online (104 f. : il., color.) : pdf.

Orientador: Eduardo Gontijo Carrano.
Coorientador: Martín Gómez Ravetti.

Tese (doutorado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Bibliografia: f.97-104.

Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Otimização multiobjetivo – Teses. 3. Consumo de energia – Teses. 4. Usinas siderúrgicas – Teses. I. Carrano, Eduardo Gontijo. II. Ravetti, Martín Gómez. III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pela Bibliotecária Letícia Alves Vieira - CRB-6/2337
Biblioteca Prof. Mário Werneck - Escola de Engenharia da UFMG

ATA DA 358ª DEFESA DE TESE DE DOUTORADO
DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

ATA DE DEFESA DE TESE DE DOUTORADO da aluna **Ana Cristina Lima Gomes** - registro de matrícula de número 2016712532. Às 08:30 horas do dia 22 do mês de março de 2021, reuniu-se na Escola de Engenharia da UFMG a Comissão Examinadora da TESE DE DOUTORADO para julgar, em exame final, o trabalho intitulado **"Matheuristics Applied To Multi-objective Production Scheduling In A Steel Industry"** da Área de Concentração em Sistemas de Computação e Telecomunicações. O Prof. Eduardo Gontijo Carrano, orientador da aluna, abriu a sessão apresentando os membros da Comissão e, dando continuidade aos trabalhos, informou aos presentes que, de acordo com o Regulamento do Programa no seu Art. 8.16, será considerado APROVADO na defesa da Tese de Doutorado o candidato que obtiver a aprovação unânime dos membros da Comissão Examinadora. Em seguida deu início à apresentação do trabalho pela Candidata. Ao final da apresentação seguiu-se a arguição da candidata pelos examinadores. Logo após o término da arguição a Comissão Examinadora se reuniu, sem a presença da Candidata e do público, e elegeu o Prof Eduardo Gontijo Carrano para presidir a fase de avaliação do trabalho, constituída de deliberação individual de APROVAÇÃO ou de REPROVAÇÃO e expedição do resultado final. As deliberações individuais de cada membro da Comissão Examinadora foram as seguintes:

Membro da Comissão Examinadora	Instituição de Origem	Deliberação	Assinatura
Prof. Dr. Eduardo Gontijo Carrano - Orientador	DEE (UFMG)	APROVAÇÃO	[illegible]
Prof. Dr. Martín Gómez Ravetti	DCC (UFMG)	APROVAÇÃO	[illegible]
Prof. Dr. Ricardo Hiroshi Caldeira Takahashi	DMAT (UFMG)	APROVAÇÃO	[illegible]
Prof. Dr. Lucas de Souza Batista	DEE (UFMG)	APROVAÇÃO	Lucas de Souza Batista.
Profa. Dra. Elizabeth Fialho Wanner	DECOM (CEFET-MG)	APROVAÇÃO	[illegible]
Prof. Dr. Thiago Henrique Nogueira	DEP (UFV)	APROVAÇÃO	Thiago H Nogueira

Tendo como base as deliberações dos membros da Comissão Examinadora a Tese de Doutorado foi aprovada. O resultado final de aprovação foi comunicado publicamente à Candidata pelo Presidente da Comissão, ressaltando que a obtenção do Grau de Doutor em ENGENHARIA ELÉTRICA fica condicionada à entrega do TEXTO FINAL da Tese de Doutorado. A Candidata terá um prazo máximo de 30 (trinta) dias, a partir desta data, para fazer as CORREÇÕES DE FORMA e entregar o texto final da Tese de Doutorado na secretaria do PPGEE/UFMG. As correções de forma exigidas pelos membros da Comissão Examinadora deverão ser registradas em um exemplar do texto da Tese de Doutorado, cuja verificação ficará sob a responsabilidade do Presidente da Banca Examinadora. Nada mais havendo a tratar o Presidente encerrou a reunião e lavrou a presente ATA, que será assinada pelo Presidente da Comissão Examinadora. Belo Horizonte, 22 de março de 2021.

ASSINATURA DO PRESIDENTE DA COMISSÃO EXAMINADORA

Acknowledgements

I would like to express my sincere gratitude to my advisor, Prof. Dr. Eduardo Carrano, for his support and embrace of my Ph.D. study and research. His guidance and shared knowledge have been of incredible help, and our meetings have always been encouraging. The notes I take from them are gold. I also appreciate and thank Prof. Dr. Martín Ravetti, my co-advisor, for all the insightful help. Our meaningful conversations challenge me to be a better professional and researcher.

To my colleagues at work, in special Marcelo Menezes, Ronaldo Sampaio, André Almeida, and Turíbio Salis: thank you for the opportunity and trust. I do not take it for granted. The investment you are making in your professionals, and the partnerships with universities, will change Industry 4.0. For my teammates, always so patient and supportive, I could not have done this without you.

My eternal love and gratefulness to my family. Mom, even though I seem too concentrated sometimes, I can always feel your pray and joy whenever I reach for something. You are my example of strength, kindness, and love. Dad, I owe you my dedication to study and research. You have always shown me how important it is. Sister, thank you for always cheering me up and making sure that non-engineers can understand the sentences I write.

My appreciation also for my beloved Arthur Farah. Your advice and shared life experiences have brought me this far. I could not be thankful enough for sharing this ride with you. I also thank my lifetime friends for all the shared moments. They come like a breath of fresh air whenever things get too heavy. To my colleagues at ORCS Lab and LaMOS, I thank you for the talks, birthday reunions, and all exchanges.

Finally, I dedicate this work to my dearest friend Lucas Criscuolo. Your memories will always live with me, and your example did not let me give up.

"It always seems impossible
until it's done"
Nelson Mandela

Abstract

Production scheduling is a challenge and opportunity for many industries seeking to improve their resource utilization and decrease their manufacturing costs. Despite that, there are still companies conducting this complex decision-making process manually, with limited business analysis. In this context, and aligned with the actual digital revolution, this research focuses on solving a scheduling problem in a multinational steel heat treatment line. A bi-objective model is proposed to minimize the line total energy costs and total tardiness. The solution is carried out through a matheuristic – a technique that combines metaheuristics and mathematical programming – which allows getting alternative solutions for production planning. A Mixed Integer Linear Programming model is designed to generate initial solutions to a Multi-objective Variable Neighborhood Search algorithm. Also, a fix-and-optimize heuristic is recommended to polish the algorithms' solutions, pulling them to the optimal front. One benefit of this proposed approach is handling large-scale problems, common in practical production scheduling cases, with reasonable computational time, alternative quality planning, and eligible scalability. The suggested matheuristic is proven to be statistically superior to the metaheuristic alone, taking as performance metric the final approximated Pareto solutions' hypervolume. Tests performed with real data from the industry showed improvements in the scheduling of the heat treatment line with reductions of energy costs and tardiness up to 14% and 100%, respectively. The methodology can also extend to other production lines of the company in the future.

Keywords: Scheduling. No-wait flow shop. Matheuristic. Multi-objective optimization. Energy consumption. Steel industry.

Resumo

O sequenciamento da produção é um desafio e uma oportunidade para muitas indústrias que buscam melhorar a utilização de seus recursos e diminuir seus custos de fabricação. Apesar disso, ainda existem empresas que conduzem esse complexo processo de tomada de decisão de forma manual, limitando as análises dos negócios. Nesse contexto, e alinhado com a atual revolução digital, esta pesquisa tem como foco a solução de um problema de sequenciamento em uma linha de tratamento térmico de uma siderúrgica multinacional. Um modelo bi-objetivo é proposto para minimizar os custos totais por consumo de energia e o tempo total de atraso na produção. A solução da formulação é realizada por meio de uma heurística matemática - técnica que combina metaheurística e programação matemática - permitindo a obtenção de soluções alternativas para o planejamento da produção. Um modelo de Programação Linear Inteira Mista é projetado para gerar soluções iniciais para um algoritmo *Variable Neighborhood Search* multiobjetivo. Além disso, uma heurística *fix-and-optimize* é apresentada para polir as soluções dos algoritmos, puxando-as para a fronteira Pareto ótima. Um benefício desta abordagem proposta é lidar com problemas de grande escala, comuns em casos práticos de programação de produção, com tempo computacional razoável, oferecendo planejamentos alternativos de qualidade e desejada escalabilidade. A heurística matemática sugerida provou ser estatisticamente superior a uma abordagem puramente metaheurística, tomando como métrica de desempenho o hipervolume final das soluções do Pareto aproximado. Testes realizados com dados reais da indústria mostraram melhorias no sequenciamento da linha de tratamento térmico com reduções de custos de energia e atrasos de até 14 % e 100 %, respectivamente. A metodologia também pode ser estendida a outras linhas de produção da empresa no futuro.

Palavras-chave: Sequenciamento. No-wait flow shop. Heurísticas matemáticas. Otimização multiobjetivo. Consumo de energia. Indústria siderúrgica.

List of figures

List of tables

List of abbreviations and acronyms

ATSP	Assymetrical Travelling Salesman Problem
AXB	Fix-and-optimize matheuristic
CB	Cooling bed
CD	Crowding distance
CL	Cooling line
EDD	Earliest Due Date
FSP	Flow shop
HF	Hardening furnace
HT	Heat treatment
HV	Hypervolume
LP	Linear programming
MH	Metaheuristics
MILP	Mixed Integer Linear Programming
MOGVNS	Multi-objective General Variable Neighborhood Search
MONWFSP	Multi-objective no-wait flow shop
MOVND	Multi-objective Variable Neighborhood Descent
MOVNS	Multi-objective Variable Neighborhood Search
MP	Mathematical programming
NG	Natural gas
NP-hard	Non-deterministic polynomial time hard
NS	Number of non-dominated solutions
NSGA-II	Non-dominated Sorting Genetic Algorithm II
NWFSP	No-wait flow shop

NWFSP-SDST	No-wait flow shop with sequence-dependent setup times
PI	Percentage improvement
QT	Quenching and tempering
RT	Runtime
SEC	Sub tour elimination constraint
SDST	Sequence-dependent setup times
SPEA2	Strength Pareto Evolutionary Algorithm
TEC	Total energy costs
TF	Tempering furnace
TSP	Travelling Salesman Problem
TT	Total tardiness
VND	Variable Neighborhood Descent
VNS	Variable Neighborhood Search
WC	Water cooling system

Contents

1 Introduction

1.1 Motivation

In 2019 the steel industry was responsible for producing 1.8 billion tons of steel (WORLD STEEL ASSOCIATION AISBL, 2020) widely used in construction, automotive industries, oil and gas sectors, and other consumer products. Despite its primary importance, manufacturers have faced challenges over the past years due to the decrease in demand and market overcapacity, forcing them to look for solutions to increase reliability, efficiency, and productivity.

In this scenario, production scheduling arises as an essential tool to promote an adequate use of the manufacturing resources. It is not easy to find the optimal allocation of people, machines, stocks, and tasks over time. The combinatorial nature of these problems includes an extensive search throughout possible solutions that, depending on the problem size, cannot be done in a feasible time, even with the use of computer algorithms. Therefore, many real scheduling problems require the use of optimization techniques.

This research is motivated by a real scheduling case in a multinational steel factory. Figure 1.1 shows the industry's production stages: ironmaking, steelmaking, continuous casting, rolling mill, heat treatment, and finishing lines. The first three processes are responsible for the iron extraction from ores, its conversion to melted steel, and later solidification through the continuous casting machine. The following rolling stage conforms the metal and adjusts its dimensions. Next, the material goes over the heat treatment for improving mechanical properties and, finally, through finishing lines to be forged, beveled, and cased in consonance with the customer's specifications. The focus of this study is the heat treatment production line and its particular scheduling goals.

The heat treatment (HT) production line is an energy-intensive process that impacts clients' delivery dates and profits for this specific industrial unit. However, it has not drawn as much attention as steelmaking and continuous casting in the scheduling literature (TANG et al., 2001; LI, 2014). For this reason, it was chosen as a pilot project with the possibility of later dissemination into lines that have similar characteristics, such as rolling mill and finishing. The research is part of a strategic Industry 4.0[1] initiative of the company that has invested in the development of global planning solutions with elaborated information systems and optimization.

[1] Industry 4.0 is a new manufacturing paradigm that empowers smart production with the adoption of intelligent equipment, Internet of Things (IoT), big data analytics, and integrated systems, as detailed in Chien, Liao, and Dou (2018). Harjunkoski et al. (2014) explains how production scheduling solutions can be a part of this industrial transformation.

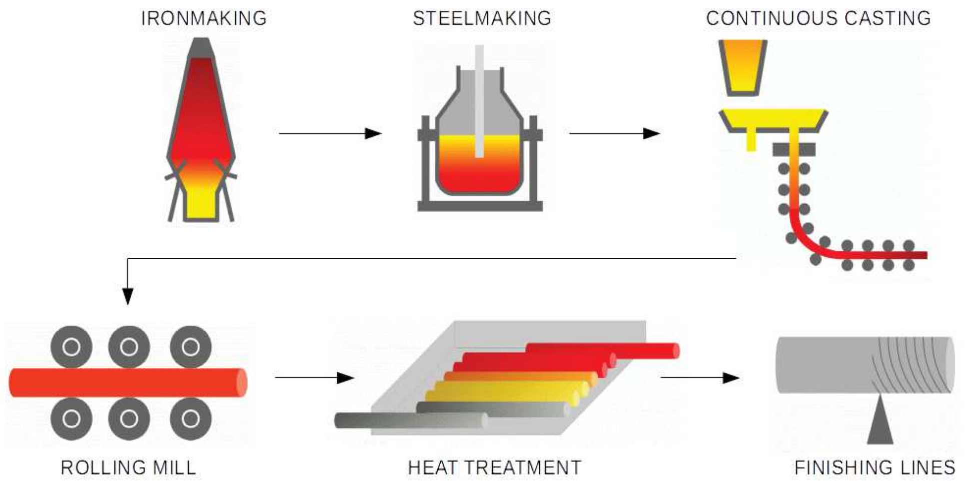


Figure 1.1 – Steel industry manufacturing processes.

The HT line is a flow shop in which four machines (three furnaces and one intermediate cooling bed with a water immersion system), set up in series, process n jobs. The process is continuous, and once the steel pieces of a job go inside the line, they flow through the machines by walking beams in the same order without interruption. Operations must follow one another immediately to avoid the degradation of the high-temperature materials and reprocessing. There are no waiting times between two successive devices. This no-wait restriction is frequent in metallurgical processes and other environments like chemical industries (ARABAMERI; SALMASI, 2013).

Products with different treatment settings require a setup to change the furnaces' temperatures. These equipment have heating rates faster than their cooling rates, so their preparation times vary according to the jobs' scheduling. When poorly planned, these sequence-dependent setup times can increase the process idle times, the total completion time, and the energy consumption.

This work aims to find scheduling solutions for the HT line modeled as a no-wait flow shop with sequence-dependent setup times (NWFSP-SDST). The goal is to minimize the production total energy costs (TEC) and total tardiness (TT).

The steel pieces in the HT line are heated in combustion furnaces powered by natural gas (NG). This fuel represents the bigger portion of changeable costs in this manufacturing line, and there is a maximum consumption per month agreed with the supplier. Minimization of this consumption can reduce production expenses, contractual fines for excessive demand, and environmental impacts. According to Zhang and Chiong (2016), investment in new equipment and hardware can contribute to industrial energy saving, but it is often costly. On the other hand, using techniques such as scheduling optimization can also promote energy saving with manufacturing reduced costs. The same concern is also present in recent steel industry applications from Hadera et al. (2015),

J. Wang et al. (2016) and Zhao, Grossmann, and Tang (2018).

The HT short-term schedules are manually calculated and often restricted to two or three days to avoid rework from unexpected events, such as machine breakdowns. The scheduling decisions are made to increase the overall equipment efficiency (reduce setup times) and meet the due dates to hand over the products to the finishing lines. A master plan defines these due dates to coordinate the manufacturing stages and accomplish the clients' final delivery requirements. A delay in the HT line can postpone finishing production and final material dispatch, which can generate contractual fines – so far, not distinguished by the client – and customer complaints.

The total energy costs and total tardiness objectives represent a trade-off, which is common in many real scheduling applications. When conflicting objectives must be achieved simultaneously, a multi-objective optimization framework is applicable. The advantage of using a multi-objective approach is presenting to the decision-makers a set of compromise solutions from which they can choose the more appropriate one (SILVA; BURKE; PETROVIC, 2004). In the HT line, this can be useful because client demands can vary over time, and the natural gas consumption can become a bigger problem at the end of the months when its supply gets closer to the contractual limits. One objective aims at the specific HT line's financial results, and the other targets a global result with a focus on client satisfaction.

In this work, we propose a multi-objective matheuristic to solve the NWFSP-SDST scheduling problem's practical variant. Matheuristics, or mathematical heuristics, are hybrid algorithms that combine heuristics and metaheuristics (MH) with mathematical programming (MP) to solve real and complex optimization problems. Metaheuristics are high-level strategies that guide non-specific problem searches to find near-optimal solutions within a reasonable time. Mathematical programming is a branch of operations research that relies on mathematical formulations and techniques to find optimal decision-making solutions. These methods ensure that the solutions found are optimal, but, in general, they are restricted to small instances, when dealing with complex and challenging problems (RAIDL; PUCHINGER, 2008). Matheuristics lie on exploring the strength of these optimization methods concerning the quality of solutions and the search time used to find them (CROCE; GROSSO; SALASSA, 2014; TALBI, 2016).

For a long time, mathematical programming was an isolated research field from metaheuristics, being studied by distant communities. However, in the last few years, researchers have recognized the potential of combining these two methods in complex problems (RAIDL; PUCHINGER, 2008). Despite its potential, few works apply matheuristics in the scope of multi-objective optimization. Ehrgott and Gandibleux (2008) cites four successful cases of this hybridization in multi-objective combinatorial optimization problems, and this author is not aware of any article or work solving the multi-objective

NWFSP-SDST problem with the use of mathematical heuristics. Therefore, this research is original.

The proposed matheuristic is a sequential execution of an exact method, a MILP heuristic, and a local search metaheuristic. In the exact method, a commercial solver is used to solve two single-objective problems. The focus is to introduce two initial solutions, one minimizing the total tardiness and the other the total energy costs, to map the Pareto front. The MILP heuristic is a fix-and-optimize approach that works as a solution polishing to improve TT and TEC. Finally, the local search metaheuristic is a Multi-objective Variable Neighborhood Search (MOVNS) algorithm, responsible for exploring the search space to map the complete approximated Pareto front.

The steel industry scheduling instances are too large (minimum of 30 jobs) to be solved exactly within a reasonable time. Thus, we propose a strategy to reduce the problem's size in the first step of the matheuristic. It provides the optimal solution in terms of energy costs, good enough cost functions for the total tardiness, and good computational time (median runtimes of around 60 seconds for each objective).

Tests conducted with real data from the steel industry indicate that the MOVNS algorithm benefits from the initial solutions generated with mathematical programming. They support the MOVNS diversity and convergence in terms of the approximated Pareto front hypervolume. Moreover, the matheuristic execution time scales well for all instances, with a complete runtime of 2.5 hours for the worst case, with almost 100 jobs.

1.2 Goals

The following general objective guides this research: mathematical heuristic development for solving a multi-objective no-wait flow shop scheduling problem with sequence-dependent setup times with application in a practical steel industry scheduling problem.

Other specific objectives are drawn to achieve the main goal:

- Study of mathematical models for the NWFSP-SDST problem and definition of a multi-objective formulation for the heat treatment total energy costs (TEC) and total tardiness (TT) minimization;
- Study and proposition of exact methods and MILP heuristics to generate and improve solutions in terms of TT and TEC;
- Suggestion of a Multi-objective Variable Neighborhood Search algorithm with suitable configurations to solve the HT scheduling problem;
- Application of the proposed matheuristic and analysis of the results over the decision-making process of the steel industry production planning.

1.3 Thesis Contributions

The primary contributions of this thesis are:

- Making a more in-depth study of the heat treatment scheduling problem for steel industries;
- Defining mathematical models for the HT problem adapted from classical no-wait flow shop formulations;
- Creating a multi-objective matheuristic general enough to be easily adapted to similar NWFSP-SDST processes;
- Designing a strategy to reduce the problems resolution size to solve them exactly, reaching energy-cost optimal solutions;
- Developing a solution polishing strategy based on MILP heuristics to improve a solution in the direction of the desired objective function;
- Demonstrating the benefits of using mathematical programming with local search metaheuristic in multi-objective real scheduling applications that require good solutions within a reasonable computational time;
- Estimating the gains of applying the methodology in the steel industry by comparing the matheuristic solutions with the real executed sequence.

1.4 Text Organization

This document has six chapters, including this introduction that presents the research motivations, specific objectives, and contributions.

The second chapter explains the heat treatment scheduling problem in detail, as it is relatively new in the scheduling literature. Examples, figures, and equations justify the problem classification and demonstrate its parameters.

The third chapter reviews two classical formulations – based on the permutation flow shop – for the NWFSP. It also reports a simplified formulation as a Traveling Salesman Problem. Finally, there is a literature survey of multi-objective NWFSP solving approaches and mathematical heuristics.

The fourth chapter details the heat treatment multi-objective model and the proposed matheuristic, describing the generation of the initial solutions, the MILP heuristic polishing, and the local search metaheuristic.

The fifth chapter shows the results of computational experiments held with data from the steel industry. Two HT scheduling formulations are compared, and the solution polishing benefits are evaluated. The designed MOVNS performance is also tested and compared to common configurations of combinatorial scheduling problems. Lastly, the expected gains for the steel industry of the proposed multi-objective matheuristic are simulated.

The final chapter synthesizes this work with a critical analysis of its results. It also suggests possible research topics for the continuity of this work.

2 The Scheduling Problem

2.1 The Heat Treatment Process

Heat treatments (HT) are tightly controlled operations in which metals are heated and cooled at specific rates to obtain particular mechanical properties, such as hardness, strength, toughness, ductility, and elasticity. There are several types of heat treatments for metals and alloys, and each of them has a specific technique and objective to make the material suitable for a particular application. Some of the different processes are normalizing, annealing, stress relieving, surface hardening, quenching, and tempering (DOSSETT; BOYER, 2006). The last two are the relevant ones in this study.

The quenching and tempering (QT) process can be divided into three steps: austenitizing, quenching, and tempering. In the first phase, the metal is heated above the steel transformation temperature starting to change its microstructure. Next, it is rapidly cooled down to achieve the desired hardiness property, the primary objective of the quenching process. Finally, in the tempering stage, the hardened steel is reheated to increase its ductility and toughness and then slowly cooled (DOSSETT; BOYER, 2006). Figure 2.1 illustrates the temperature curve of a material during a QT treatment.

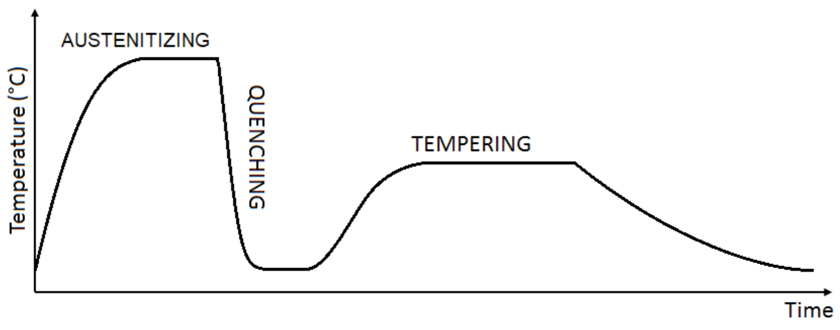


Figure 2.1 – Desired temperature profile during quenching and tempering.

(Redrawn from source: tec-science (2018))

Figure 2.2 shows the layout of the studied QT line. The process begins in the hardening furnace (HF), also known as the austenitizing furnace. Immediately after the material is heated, each steel piece goes into a water cooling system, where it is rapidly cooled down. After the water immersion, the material goes through a cooling bed (CB) and then enters the tempering furnaces (TF1 and TF2). Finally, the material is slowly cooled down by still air in the subsequent cooling line (CL).

The furnaces and cooling beds are scaled for high production, and they are capable of treating more than one steel piece with the same process conditions simultaneously, as

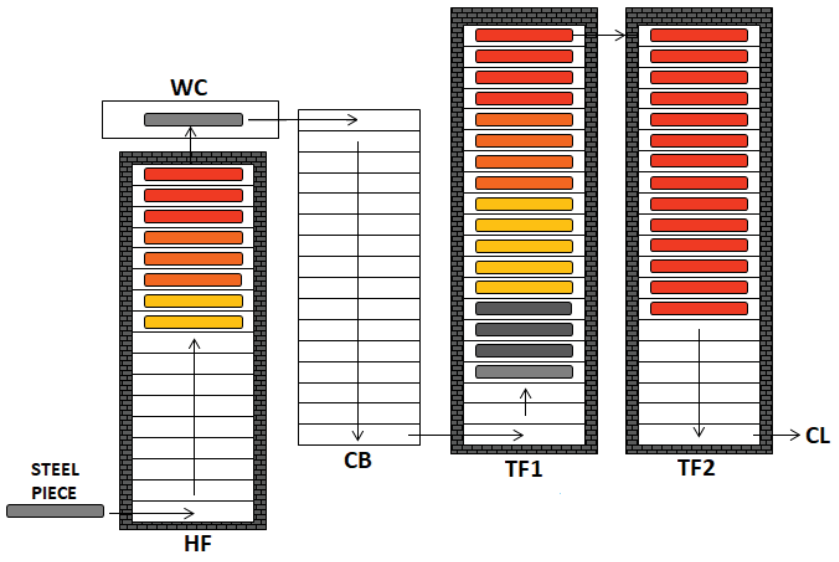


Figure 2.2 – Schematic of the heat treatment production line.

The colored rectangles represent the steel pieces that move through the line walking beams (white rectangles) following the directions of the arrows. The gray rectangles exemplify cold pieces that turn red once they are heated up. The presented machines are hardening furnace (HF), water cooling system (WC), cooling bed (CB), and tempering furnaces (TF1 and TF2). After the last furnace, the pieces leave to a cooling line (CL).

in batch processes. In Figure 2.2, there are two jobs in the process. The first one has 31 pieces in the tempering furnaces, and the second one has nine pieces in the hardening furnace and water cooling system. Each machine has its capacity, which is the number of walking beams inside the equipment (each beam moves only one steel piece at a time). The white rectangles of Figure 2.2 represents the walking beams.

In the steel industry quenching and tempering line, a client request for some steel pieces of a particular product generates a production job with specific process parameters, that are: production flow rate for each piece and treatment temperatures in the HF and TF furnaces. The combination of these parameters establishes the desirable heating and cooling curves of the treated material, as presented in Figure 2.1. A steel piece is reprocessed if it does not comply with the adequate curve.

The discrete process is without interruption, and once a material goes inside the HF, it flows through the machines in the same order without interruption, according to the arrows shown in Figure 2.2. Operations must follow one another immediately to avoid the degradation of the high-temperature materials. There are no waiting times between two successive devices. This no-wait restriction is frequent in metallurgical processes and other environments like chemical industries (ARABAMERI; SALMASI, 2013).

Products with different treatment settings require a line setup, which is needed to change the furnaces temperatures. These equipment have heating rates faster than their cooling rates, so their preparation times vary according to the jobs' scheduling. When poorly planned, these sequence-dependent setup times can increase the process idle times, the total completion time, and the energy consumption.

2.2 Scheduling in the Heat Treatment

In this work, the HT line is modeled as a no-wait flow shop with sequence-dependent setup times (NWFSP-SDST). The following sections detail the problem classification and scheduling particularities.

2.2.1 Problem Classification

Scheduling problems are described and distinguished in the literature by a triplet $\alpha|\beta|\gamma$. This notation was proposed by Graham et al. (1979) to represent, respectively, its machine environment, processing characteristics, and optimality criteria. The machine environment α is normally expressed by an uppercase letter and the number of machines. The β field can have multiple entries to show different aspects of the process such as permutation, precedence constraints, no-wait restrictions, and sequence-dependent setup times. The γ field specifies the scheduling objectives, which can be one or more.

The HT scheduling problem three field representation is $F_4|nwt, s_{irk}|TT, TEC$: a flow shop of four machines (F_4) with no-wait (nwt) and sequence-dependent setup times (s_{irk}) restrictions to minimize the production total tardiness (TT) and total energy costs (TEC).

A flow shop (FSP) is a well-known machine environment, typical in many manufacturing facilities, in which m machines are set up in series, and they process n jobs following the same route (PINEDO, 2012). Figure 2.3 illustrates in a Gantt chart[1] the progress of two jobs in a three-machine flow shop in which it can be seen that the k^{th} operation of a job is performed in the k^{th} machine, and for each pair of job and machine there is a specific processing time.

A no-wait flow shop (NWFSP) is a particular case of a FSP where the operations of a job from machine 1 to m cannot be interrupted, and there are no waiting times between two successive devices. In Figure 2.3 it is possible to see that the second job ($J2$) has to wait for the availability of machine three ($M3$) after processing in machine two ($M2$). This cannot happen in the quenching and tempering line because the steel would lose its temperature, harming its mechanical properties. Therefore, in the no-wait scenario, additional time has to be added before the start of the second job in the first machine ($M1$) so that all of its operations can continuously follow each other.

[1] A Gantt chart is a type of bar chart used to illustrate a schedule. The vertical axis displays the tasks and the horizontal axis the time intervals. The width of the parallel bars shows the duration of each activity.

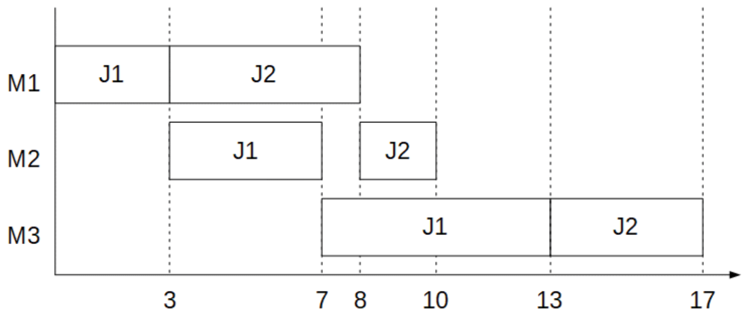


Figure 2.3 – Flow shop Gantt chart.

Figure 2.4 exemplifies the effect of the no-wait constraints in the flow shop Gantt chart, in which three units of time delay the start of the second job. This characteristic is frequent in many practical cases (plastic production, chemical industries, and metallurgical processes), in which the wait between machines can damage the products, and it also appears in just-in-time manufacturing, which does not maintain inventories between the stages of production (NAGANO; MIYATA, 2016). Another singularity of a NWFSP is that the order of the tasks for all machines is the same, so the first material that goes into production is also the first to get out (PINEDO, 2012). This last feature is called permutation flow shop.

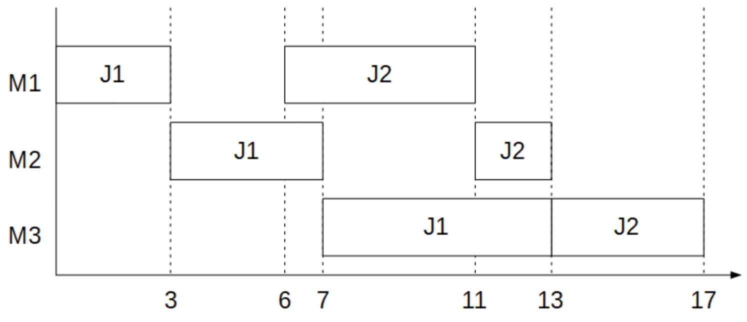
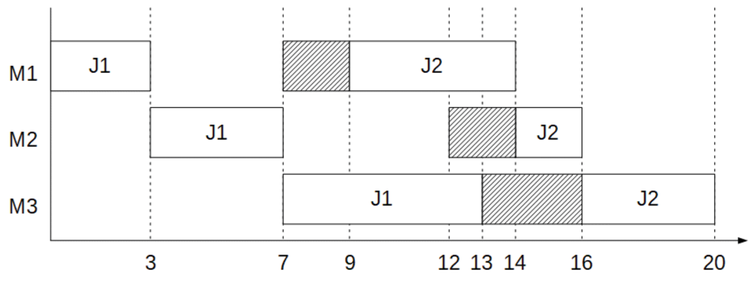


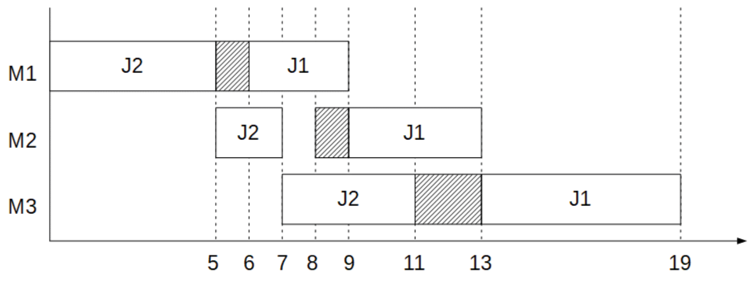
Figure 2.4 – Flow shop Gantt chart with no-wait constraints.

Another common feature in manufacturing processes is the preparation time for each machine, also known as setup time. This time can be used to change a piece of equipment to treat a new product, adjust them with new device parameters, among other activities. They are classified as sequence-independent or sequence-dependent, in which the preparation time of a job changes according to what has been processed before. The latest is the case of the HT scheduling problem because it takes longer to setup up the furnaces' temperature from a material with a high temperature to a low one than otherwise.

Sequence-dependent setup times (SDST) interferes with line productivity and, when poorly planned, can increase the process idle times and total completion time. This is demonstrated in Figure 2.5 that includes SDST (hashed rectangles) in the NWFSP shown in Figure 2.4. The setups from $J1$ to $J2$, shown in 2.5a, are longer than the setups from $J2$ to $J1$, shown in 2.5b. Therefore, the idle times and the completion time of the jobs in the machines are bigger in the first scenario. Because of its importance and impact, in the mid-1960s, SDST were explicitly considered in production scheduling researches, mainly driven by industrial and services applications (ALLAHVERDI, A., 2015).



(a) Setups from job 1 to job 2.



(b) Setups from job 2 to job 1.

Figure 2.5 – Effect of sequence-dependent setup times in a no-wait flow shop Gantt chart.

2.2.2 Scheduling Parameters

A job i in the QT line represents a set of steel pieces, y_i, requested by a client. These pieces require the same process settings: the furnaces temperatures, t_{ik}, and the walking beam cycle time, c_i (given in seconds), which is the same in all the m machines (the material speed is constant during the whole process). For each job, i, there is a due date, d_i, which represents the maximum date the next production stage expects to receive the treated steel pieces. Table 2.1 illustrates these process parameters in a five jobs fictitious instance. These values were created to clarify the process and do not represent the study case reality, in order to protect company sensitive information. The temperatures of columns 4 to 7 were defined based on the QT typical ranges presented in Dossett and Boyer (2006).

Figure 2.6 shows a schedule view of the heat treatment line for the fictitious instance of Table 2.1. The jobs are sequenced in the same order they appear in the table. The x-axis, on the top of the graph, displays the machines with a scale band proportional to the machine capacity. The hours are on the vertical axis, and the parallelograms represent the five consecutive jobs.

Table 2.1 – Fictitious instance with five jobs for demonstration of the QT scheduling problem.

i	y_i	$c_i[s]$	$t_{i1}[°C]$	$t_{i2}[°C]$	$t_{i3}[°C]$	$t_{i4}[°C]$	d_i
1	45	40	825	0	300	400	1
2	90	20	850	0	325	475	1
3	70	20	850	0	325	475	2
4	60	30	850	0	325	475	2
5	30	30	815	0	325	475	2

The triangles presented in the HF quadrant and over the first job represent the time the first steel piece entered and left the hardening furnace. The circles represent the time the last steel piece of that job entered and left the same furnace. In traditional batching processes and Gantt charts, a job has a unique starting and ending time in a machine (e.g. Figure 2.5). For the QT line, there are two starting and two ending times.

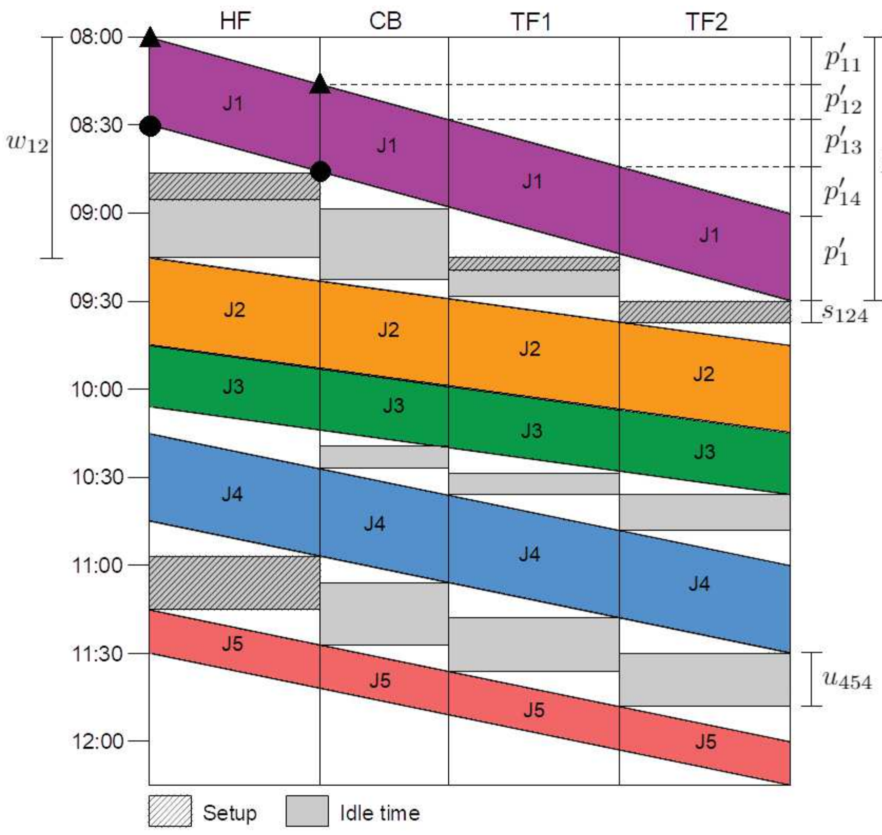


Figure 2.6 – Schedule view of the heat treatment with a demonstration of its parameters.

Each machine appears on the horizontal axis, with the width proportional to its capacity, and the time is on the vertical axis. The parallelograms represent five consecutive jobs, respectively, $J1$, $J2$, $J3$, $J4$, and $J5$ (Table 2.1). The hashed rectangles show the machines setup times between the jobs, and the gray rectangles are the total idle times.

The gray rectangles represent the idle times on the machines, and the hashed rectangles the setup times. Four scenarios for two consecutive jobs are exemplified in Figure 2.6. The first scenario is between $J1$ and $J2$, which have different treatment temperatures for all furnaces and different cycle times. As $J2$ is faster than $J1$ (shorter cycle time), waiting time is needed in machine 1 to guarantee that the first steel pieces of $J2$ will arrive in the TF2 furnace only after the last steel piece of $J1$ left and the temperature changed. The second scenario is between $J2$ and $J3$. They have the same cycle time and temperatures for all furnaces, being considered as in the same job family. There is no need to wait for the last piece of a job to leave the furnace. Their steel pieces can go into the machines one after the other without waiting times. In these cases, two different jobs can be in the same machine at the same time. It reduces the production line overall energy costs and completion time. The third scenario is when there are no changes in the furnaces temperatures, but there is a change in the cycle time, as it happens for $J3$, followed by $J4$. The first steel piece of $J4$ can only enter the furnace after the $J3$ is done. As job $J4$ is slower, this leads to idle times in the TF furnaces. Finally, if the cycle time is the same, but at least one treatment temperature is different, there will be idle times in the machines, as we can see in the sequence $J4$-$J5$.

A job processing time through the entire line, p_i, is the total hours between the first steel piece that goes into production and the last one that goes out:

$$p_i = p'_i + \sum_{k=1}^{m} p''_{ik}, \tag{2.1}$$

in which $p'_i = c_i y_i/3600$ represents the total time spent to enter (or exit) all the y_i pieces to be produced in job i in the production line. It depends on the production rate (walking beams cycle time) of the material, c_i, and defines the height of the job parallelogram in the QT schedule view. All the process times are in hours; thus, the division by 3600. The processing time of the last steel piece of the job - time between its charge in machine 1 and its discharge in machine m -, $p''_{ik} = c_i q_k/3600$, is proportional to the inclination of the parallelograms of Figure 2.6, and q_k is the capacity of machine k.

Each furnace has its own setup time, s_{irk}, that depends on its cooling and heating rates (respectively, r^c_k and r^h_k in $°C/h$) and the difference in the consecutive jobs, i and r, treatment temperatures:

$$s_{irk} = \begin{cases} (t_{ik} - t_{rk})/r^c_k & \text{if } t_{ik} > t_{rk} \\ (t_{rk} - t_{ik})/r^h_k & \text{if } t_{ik} < t_{rk} \\ 0 & \text{if } t_{ik} = t_{rk} \end{cases}. \tag{2.2}$$

The furnaces' cooling rates are smaller than the heating rates, therefore, the condition of $t_{ik} > t_{rk}$ causes longer setup times.

Equation 2.3 denotes the necessary waiting time between the start of two consecutive jobs, i and r, in the first machine, to guarantee the no-wait restrictions. It is based on the no-wait flow shop transformation into a traveling salesman problem (TSP) that will be further reviewed in the next chapter.

$$w_{ir} = p'_i + \max_{k \in \{1,\ldots,m\}} \left(\sum_{l=1}^{k} p''_{il} + s_{irk} - \sum_{l=1}^{k-1} p''_{rl}, 0 \right). \tag{2.3}$$

If two consecutive jobs have the same process settings - that require neither setups nor idle times - they can be produced one immediately after the other; therefore, $w_{ir} = p'_i$.

In the course of production, the furnaces regularly burn natural gas (NG). A furnace consumes NG constantly while processing the steel pieces from a job and also when it is empty, preparing itself for the entrance of new material. The time each furnace k stays idle, between two consecutive jobs i and r, is

$$u_{irk} = w_{ir} + \sum_{l=1}^{k-1} p''_{rl} - \left(p'_i + \sum_{l=1}^{k} p''_{il} \right). \tag{2.4}$$

The total NG volume consumed by job i, given in Nm^3, is sequence-dependent and depends on two terms. The first one is the total volume consumed during the material processing,

$$v'_i = \sum_{k=1}^{m} (p'_i + p''_{ik}) f_{ik}, \tag{2.5}$$

which depends on the constant NG flow, f_{ik}, required to keep the furnace at temperature t_{ik}. The second term is the volume consumed during the idle times,

$$v''_{ir} = \sum_{k=1}^{m} u_{irk} f_{irk}, \tag{2.6}$$

which depends on the NG flow required to change the furnace temperature, f_{irk}, from t_{ik} to t_{rk}. This flow is smaller when there is a decrease in the furnaces' temperatures $(t_{ik} > t_{rk})$. As a consequence, schedules with lower energy consumption may have longer setup times harming its total tardiness.

The scheduling parameters presented here are important for understanding the HT mathematical models that will be introduced in Chapter 4. The next chapter reviews mathematical formulations and multi-objective solving approaches for the no-wait flow shop problem. They will allow a better comprehension of this work proposed methodology and its contributions.

3 Literature Review

3.1 Mathematical models for No-Wait Flow Shop Problems

This chapter begins with a literature review of mathematical models that can represent the no-wait flow shop (NWFSP) problems. It will start with an overview of flow shop formulations that will allow the evaluation of different variables to address the problem. Next, these models will be extended to engage the no-wait restrictions bounded by its representation as a traveling salesman problem (TSP).

The mathematical models presented here will focus on the total tardiness (TT) and makespan (C_{max}) objectives, that are extensively explored in the literature. The proposed multi-objective model for the heat treatment scheduling problem, with the total energy costs objective, will be presented in Chapter 4, based on the following review.

3.1.1 Overview of Permutation Flow Shop Formulations

As mentioned in Chapter 2, the NWFSP is a variant from the permutation FSP problem. In the literature, there are different formulations for the exact solution of permutation FSP. According to Pan (1997) and Tseng, Stafford, and Gupta (2004), the three models that better represent this class of problems are from Wagner (1959), Wilson (1989), and Manne (1960). The first two are appointed as positional-based models due to their assignment of jobs in the sequence through the binary variable z_{ij}, which is equal to 1 if job i goes into position j of schedule or 0 otherwise. The last one is known as a precedence-based model as it uses a binary variable x_{ir} to establish the precedence relation between two jobs, that is, x_{ir} is equal to 1 if job i is scheduled before job r or 0 otherwise.

The three models are described next, and the indices and parameters used by them are:

i, r	- indices for jobs,
j	- index for position in sequence,
k	- index for machine,
n	- number of jobs,
m	- number of machines,
d_i	- due date of job i,
p_{ik}	- processing time of job i in machine k,
M	- a very large constant.

The total tardiness objective taken into consideration in this section is calculated by

$$T_i = \max\left(C_{im} - d_i, 0\right), \tag{3.1}$$

in which C_{im} is the completion time of job i in the last machine m.

The first model proposed by Wagner (1959) is an all integer formulation for the solution of a three machine FSP. Later, Stafford (1988) suggested some improvements for environments with more than three machines and transformed the model in a mixed formulation. Variables of the improved model are:

z_{ij}	- 1 if job is assigned to position j of sequence or 0 otherwise,
I_{jk}	- idle time on machine k between jobs in sequence position j and $j+1$,
W_{jk}	- waiting time of job in sequence position j between machines k and $k+1$,
C_{jk}	- completion time of job in sequence position j at machine k,
T_j	- tardiness of job in sequence position j.

The original formulation is based on the minimization of the makespan. Below is the formulation of the model for a $F_m|prmu|\sum T_j$, as shown in Ronconi and Birgin (2012).

$$\text{Min } \sum_{j=1}^{n} T_j \tag{3.2}$$

$$\text{s.t. } \sum_{j=1}^{n} z_{ij} = 1 \qquad \forall\, i \in \{1,\dots,n\}, \tag{3.3}$$

$$\sum_{i=1}^{n} z_{ij} = 1 \qquad \forall\, j \in \{1,\dots,n\}, \tag{3.4}$$

$$I_{jk} + \sum_{i=1}^{n} z_{i,j+1} p_{ik} + W_{j+1,k} = W_{jk} + \sum_{i=1}^{n} z_{ij} p_{i,k+1} + I_{j,k+1} \qquad \forall\, j \in \{1,\dots,n-1\}, k \in \{1,\dots,m-1\}, \tag{3.5}$$

$$W_{jk} \geq 0 \qquad \forall\, j \in \{1,\dots,n\}, k \in \{1,\dots,m-1\}, \tag{3.6}$$

$$I_{jk} \geq 0 \qquad \forall\, j \in \{1,\dots,n-1\}, k \in \{1,\dots,m\}, \tag{3.7}$$

$$C_{1m} = \sum_{k=1}^{m-1} \left(\sum_{i=1}^{n} z_{i1} p_{ik} + W_{1k} \right) + \sum_{i=1}^{n} z_{i1} p_{im}, \tag{3.8}$$

$$C_{jm} = C_{j-1,m} + I_{j-1,m} + \sum_{i=1}^{n} z_{ij} p_{im} \qquad \forall\, j \in \{2,\dots,n\}, \tag{3.9}$$

$$T_j \geq C_{jm} - \sum_{i=1}^{n} z_{ij} d_i \qquad \forall\, j \in \{1, \ldots, n\}, \tag{3.10}$$

$$T_j \geq 0 \qquad \forall\, j \in \{1, \ldots, n\}, \tag{3.11}$$

$$z_{ij} \in \{0, 1\} \qquad \forall\, i, j \in \{1, \ldots, n\}. \tag{3.12}$$

Equation 3.2 represents the total tardiness minimization objective. Constraints 3.3 and 3.4 are classic from assignment problems and ensure that job i is assigned to only one position in schedule and that each position j has only one job assigned to it. Constraint set 3.5 ensures that job j cannot be processed in machine $k+1$ until it has finished processing in machine k and that machine k cannot process the job $j+1$ until it has finished job j. Figure 3.1 exemplifies this constraint and shows the relationship between processing times, idle times, and waiting times. The last two must be non-negative, hence, there are constraint sets 3.6 and 3.7. The completion time of all the n jobs in the last machine m is determined by constraint 3.8 and constraint set 3.9. The total tardiness, shown in equation 3.1, of the job allocated in sequence position j is linearized at constraints 3.10 and 3.11. Finally, constraint set 3.12 guarantees the binary integrity of the positional variable z_{ij}.

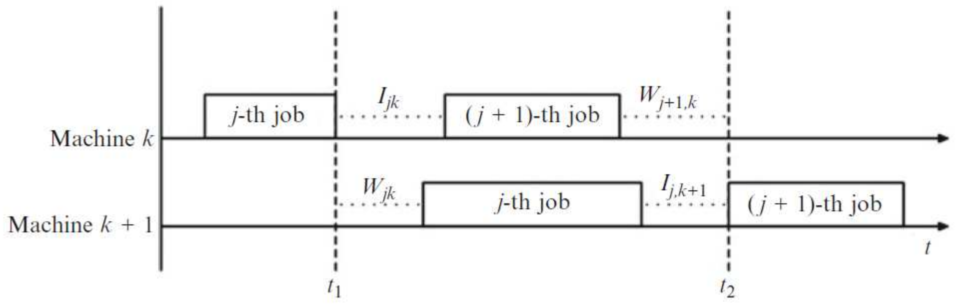


Figure 3.1 – Graphical representation of Wagner's model constraint set 3.5. (RONCONI; BIRGIN, 2012).

The second positional model considered for the solution of the permutation FSP was proposed by Wilson (1989) and is based on Wagner's formulation. Instead of dealing with idle and waiting times, Wilson introduced a continuous start time variable S_{jk} for each job in position j and each machine k. Wilson's model is obtained by removing constraints 3.5, 3.6, 3.7, 3.8, and 3.9 of Wagner's model and including constraints for the start time variable, as shown below.

$$\text{Min } \sum_{j=1}^{n} T_j \tag{3.2}$$

$$\text{s.t. } \sum_{j=1}^{n} z_{ij} = 1 \qquad \forall\, i \in \{1,\dots,n\}, \tag{3.3}$$

$$\sum_{i=1}^{n} z_{ij} = 1 \qquad \forall\, j \in \{1,\dots,n\}, \tag{3.4}$$

$$S_{11} \geq 0, \tag{3.13}$$

$$S_{j,k+1} \geq S_{jk} + \sum_{i=1}^{n} z_{ij}p_{ik} \qquad \forall\, j \in \{1,\dots,n\}, k \in \{1,\dots,m-1\}, \tag{3.14}$$

$$S_{j+1,k} \geq S_{jk} + \sum_{i=1}^{n} z_{ij}p_{ik} \qquad \forall\, j \in \{1,\dots,n-1\}, k \in \{1,\dots m\}, \tag{3.15}$$

$$C_{jm} = S_{jm} + \sum_{i=1}^{n} z_{ij}p_{im} \qquad \forall\, j \in \{1,\dots,n\}, \tag{3.16}$$

$$T_j \geq C_{jm} - \sum_{i=1}^{n} z_{ij}d_i \qquad \forall\, j \in \{1,\dots,n\}, \tag{3.10}$$

$$T_j \geq 0 \qquad \forall\, j \in \{1,\dots,n\}, \tag{3.11}$$

$$z_{ij} \in \{0,1\} \qquad \forall\, i,j \in \{1,\dots,n\}. \tag{3.12}$$

Constraint 3.13 states that the start time of the first job of the sequence in the first machine is at least zero. Constraint set 3.14 guarantees that a job only starts in machine $k+1$ after finishing its process in machine k and constraints 3.15 ensures that machine k only starts processing the job in position $j+1$ after finishing the previous j job. At last, constraint set 3.16 shows that the completion time of a job in the last machine is the start of that job in the machine plus its processing time.

The third model considered for the solution of the permutation FSP was proposed by Manne (1960) to solve job shop scheduling problems for minimization of makespan. Variables of this model are:

x_{ir}	- 1 if job i is scheduled any time before job r or 0 otherwise,
C_{ik}	- completion time of job i at machine k,
T_i	- tardiness of job i.

Below is its formulation for the minimization of total tardiness, also based on Ronconi and Birgin (2012).

$$\text{Min } \sum_{i=1}^{n} T_i \tag{3.17}$$

$$\text{s.t. } C_{i1} \geq p_{i1} \qquad \forall\, i \in \{1,\dots,n\}, \tag{3.18}$$

$$C_{ik} \geq C_{i,k-1} + p_{ik} \qquad \forall\, i \in \{1,\dots,n\}, k \in \{2,\dots,m\}, \tag{3.19}$$

$$C_{ik} \geq C_{rk} + p_{ik} - Mx_{ir} \qquad \forall\, i,r \in \{1,\dots,n\}, k \in \{1,\dots,m\}, \tag{3.20}$$

$$C_{rk} \geq C_{ik} + p_{rk} - M(1 - x_{ir}) \qquad \forall\, i,r \in \{1,\dots,n\}, k \in \{1,\dots,m\}, \tag{3.21}$$

$$T_i \geq C_{im} - d_i \qquad \forall\, i \in \{1,\ldots,n\}, \tag{3.22}$$
$$T_i \geq 0 \qquad \forall\, i \in \{1,\ldots,n\}, \tag{3.23}$$
$$x_{ir} \in \{0,1\} \qquad \forall\, i,r \in \{1,\ldots,n\}. \tag{3.24}$$

Total tardiness minimization is expressed in equation 3.17 and, different from the positional models, it is indexed by the job itself. Constraint set 3.18 secures that the completion time of job i in the first machine is at least its processing time in that machine and constraints 3.19 guarantee that the completion time of job i in machine k is at least the completion time of the job in the machine before and its processing time in machine k. Constraint sets 3.20 and 3.21 are disjunctive constraints that insures job i either precedes job r or follows it and its completion time is dependent on the completion time of the precedence job. Linearization of the tardiness calculation by job appears in constraints 3.22 and 3.23. Finally, constraint set 3.24 guarantees the binary integrity of precedence variable x_{ir}.

In a first analysis, Manne's model was considered by Pan (1997) as the best formulation to solve the permutation FSP because it has the least number of binary variables. However, Tseng, Stafford, and Gupta (2004) showed that the computational time of positional models overcomes the precedence one due to the absence of disjunctive inequalities that enlarge matrix sizes and makes the set of constraints less tighten. The second conclusion of this study was that Wagner's model performed better than Wilson's for having fewer constraints. The paper from Tseng, Stafford, and Gupta (2004) focuses in the minimization of makespan, but same results were found by Ronconi and Birgin (2012) for the total tardiness objective. Naderi et al. (2012) also showed that positional models have more performance in the solution of a multi-objective no-wait flow shop problem.

3.1.2 Formulation as a Traveling Salesman Problem

In the literature, both positional and precedence models appear on the resolution and representation of no-wait flow shop problems. Chen, Wang, and Peng (2019), Samarghandi and ElMekkawy (2014), and Liu, Song, and Wu (2013) present positional models to solve, respectively, the NWFSP for the minimization of energy consumption, makespan, and total tardiness. Differently, Komaki and Malakooti (2017) and Lin and Ying (2016) propose precedence models to solve the minimization of makespan. What these papers – and the majority of the NWFSP researches – have in common is that the problem formulation relies on its representation as a special case of the traveling salesman problem (TSP).

The TSP is a well-known combinatorial optimization problem studied in operations research. Given a set of n cities, the problem consists of finding the minimum distance tour that visits each city once, starting and ending at the same point. The problem can be classified into symmetric when the distance between two cities is independent of the direction taken, and asymmetrical when the distance depends on the direction. Figure 3.2 exemplifies an asymmetrical traveling salesman problem (ATSP) graph with four cities. The circles represent the cities and the directed arcs the distance between them. In this case, considering the start in city 1, the optimal tour, highlighted in red, is the one that visits the cities in the following order: 1, 3, 4, 2, 1. The ATSP applies to many real logistics and scheduling problems. In the latter, it can be used to solve single machine problems with sequence-dependent setup times and no-wait flow shop problems, both with a makespan objective (CAMPUZANO; OBREQUE; AGUAYO, 2020).

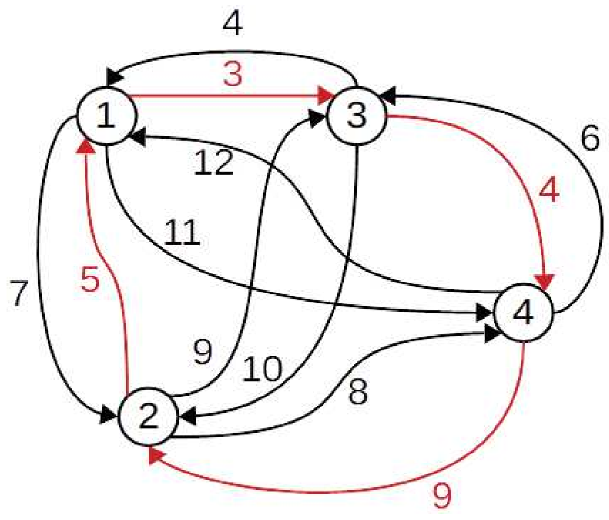


Figure 3.2 – Example of an ATSP graph with four cities and the minimum distance tour highlighted in red.

The ATSP is typically formulated as an integer problem with the binary variable y_{ir}, which indicates whether the path from city i to r, represented by the $arc(i,r)$, is part of the tour. Below is the basic formulation, in which $[c_{ir}]$ is the asymmetric intercity distance matrix.

$$\text{Min} \sum_{i,r=1}^{n} y_{ir}c_{ir} \tag{3.25}$$

$$\text{s.t.} \sum_{i=1}^{n} y_{ir} = 1 \qquad \forall\, r \in \{1,\dots,n\}, \tag{3.26}$$

$$\sum_{r=1}^{n} y_{ir} = 1 \qquad \forall\, i \in \{1,\dots,n\}, \tag{3.27}$$

$$SECs \tag{3.28}$$

$$y_{ir} \in \{0,1\} \qquad \forall\, i,r \in \{1,\dots,n\}. \tag{3.29}$$

Equation 3.25 is the objective function that minimizes the total distance traveled between the cities. Constraint sets 3.26 and 3.27 are the assignment constraints, which ensure that each node is visited and departed only once. Constraints 3.28 are known as the sub tour elimination constraints (SECs) to avoid isolated loops in the tour. Constraint sets 3.29 guarantees the binary integrity of variable y_{ir}.

There are several SECs formulations for the traveling salesman problem, as shown by Öncan, Altınel, and Laporte (2009) in his extensive survey over the asymmetrical case. Two of the classical representations are from Dantzig, Fulkerson, and Johnson (1954) and Miller, Tucker, and Zemlin (1960), known, respectively, as the DFJ and MTZ formulations. The latter was proposed based on the Vehicle Routing Problem and will be the one applied in this work. The choice was based on its polynomial-length and compact formulation.

The MTZ formulation uses the continuous variable u_i to enumerate the order in which the cities are visited in the tour. The constraints are:

$$u_i - u_r + ny_{ir} \leq n - 1 \quad \forall i, r \in \{1, \dots, n\}, \tag{3.30}$$

$$1 \leq u_i \leq n - 1 \quad \forall i \in \{1, \dots, n\}. \tag{3.31}$$

The first time the NWFSP was noticed as a special case of the ATSP was by Piehler (1960) in the minimization of makespan (MUCHA; SVIRIDENKO, 2013), which was later reinforced by Wismer (1972). They defined a waiting time parameter, w_{ir}, that represents the minimum delay necessary between the start of two consecutive jobs, i and r, in the first machine. This time guarantees the continuous flow of job r through all the machines when job i immediately precedes it.

Figure 3.3 shows the Gantt chart of three jobs and three machines no-wait flow shop. The hashed rectangles are the waiting times between jobs $J1$-$J2$ and $J2$-$J3$. From the example, it is possible to see that $J2$ must wait 6 units of time after the start of $J1$ to go into machine 1. Otherwise, machine 3 will not be available when $J2$ finishes in the second machine and there will be an interruption in its production. On the contrary, $J3$ can enter machine 1 right after $J2$, so its waiting time is only the processing time of $J2$ in machine 1. This parameter has also been exemplified in Figure 2.6 for the HT scheduling particular case.

Generalizing, the waiting time between two jobs in a NWFSP can be calculated by

$$w_{ir} = p_{i1} + \max_{k \in \{1, \dots, m\}} \left(\sum_{q=1}^{k} p_{iq} - \sum_{q=1}^{k-1} p_{rq}, 0 \right), \tag{3.32}$$

where p_{ik} is the process time of job i in machine k.

The former equation can also be modified to consider sequence-dependent setup times, s_{irk}, as shown in the works from Bianco, Dell'Olmo, and Giordani (1999), and

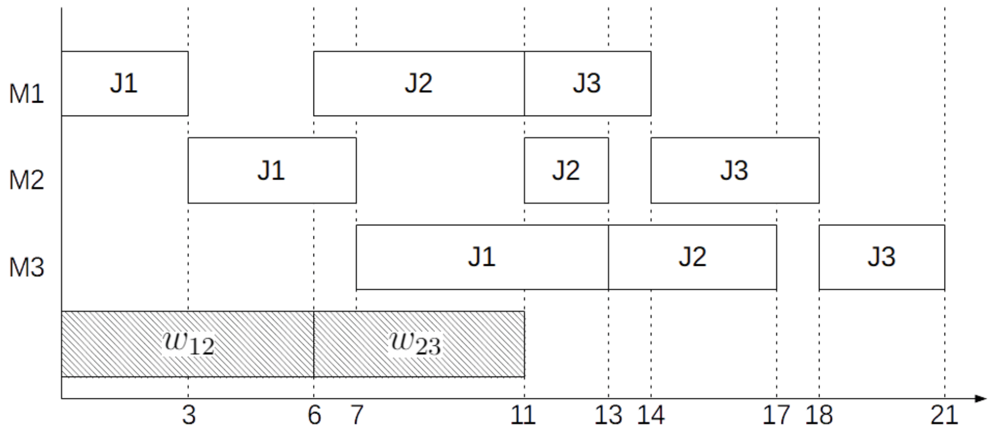


Figure 3.3 – Gantt chart of a three jobs and three machines NWFSP with its waiting times.

Nagano and Araújo (2014):

$$w_{ir} = p_{i1} + \max_{k\in\{1,\ldots,m\}} \left(\sum_{q=1}^{k} p_{iq} + s_{irk} - \sum_{q=1}^{k-1} p_{rq}, 0 \right). \tag{3.33}$$

Equation 3.33 supported the definition of the heat treatment waiting time parameter presented in Chapter 2. The main difference is that Equation 2.3 takes into consideration the two processing times (p'_i and p''_{ik}) of our real scheduling case.

From Figure 3.3, it can be observed that the makespan of sequence 1, 2, and 3 is equal to the sum of the waiting times with the total processing time of $J3$ in all machines $(6+5+4+3+3=21)$. Therefore, introducing a dummy job $n+1$, that indicates the start and end of the sequence, and defining

$$w_{i,n+1} = \sum_{k=1}^{m} p_{ik}, \tag{3.34}$$

and

$$w_{n+1,i} = 0, \tag{3.35}$$

the makespan of the sequence can be calculated by

$$C_{max} = \sum_{i,r=1}^{n+1} y_{ir} w_{ir}, \tag{3.36}$$

in which variable y_{ir} is equal to 1 when job i immediately precedes job r. This variable is different from Manne´s x_{ir} variable due to the definition of the immediate precedence.

3.1.2.1 Precedence-based Model

Considering the waiting time matrix as the intercity distances matrix, and given a new set of jobs, $n' = n+1$, the $F_m|nwt|C_{max}$ and $F_m|nwt, s_{irk}|C_{max}$ problems can be formulated as an ATSP:

$$\begin{aligned}
\text{Min} \quad & \sum_{i,r=1}^{n'} y_{ir} w_{ir} && && (3.37)\\
\text{s.t.} \quad & \sum_{r=1}^{n'} y_{ir} = 1 && \forall\, i \in \{1,\dots,n'\}, && (3.38)\\
& \sum_{i=1}^{n'} y_{ir} = 1 && \forall\, r \in \{1,\dots,n'\}, && (3.39)\\
& u_i - u_r + n' y_{ir} \leq n' - 1 && \forall\, i,r \in \{2,\dots,n'\}, && (3.40)\\
& 1 \leq u_i \leq n' && \forall\, i \in \{1,\dots,n'\}, && (3.41)\\
& y_{ir} \in \{0,1\} && \forall\, i,r \in \{1,\dots,n'\}. && (3.42)
\end{aligned}$$

Equation 3.37 represents the minimization of the makespan objective. Constraint sets 3.38, 3.39, and 3.42 are the same assignment constraints from ATSP, and the subtour elimination constraints are given by 3.40 and 3.41.

The model presented above is used for the solution of the makespan. However, the waiting time parameter can also support a NWFSP positional-based formulation for the minimization of the total tardiness.

3.1.2.2 Positional-based Model

Through equation 3.32 it is possible to define the relationship between the start of two consecutive jobs in the first machine of the NWFSP. Considering that job r is processed after job i and that the start time of job i in the first machine (S_{i1}) is known, the start time of job r in machine 1 is determined by

$$S_{r1} \geq S_{i1} + w_{ir}. \tag{3.43}$$

Since the operations of a job in a no-wait flow shop are uninterrupted, from the start time of job r in machine 1 it is possible to determine the completion time of this job in the last machine:

$$C_{rm} = S_{r1} + \sum_{k=1}^{m} p_{rk}. \tag{3.44}$$

Equation 3.44 can then be simplified where p_r is the sum of all the processing times of job r in all the m machines, S_r is the start time of job r in the first machine and C_r is the completion time of the same job in the last machine:

$$C_r = S_r + p_r. \tag{3.45}$$

Based on the equations above, a simplified formulation is proposed for the solution of the $F_m|nwt, s_{irk}|\sum T_j$. It takes into consideration the model proposed by Wilson, due to the direct association of the waiting time calculation with the start time continuous variable. A new variable, b_{irj}, is added to define the precedence relationship between two jobs i and r. These variables are necessary due to the no-wait and sequence-dependent setup times restrictions of the problem, as can be seen in the studies of Stafford and Tseng (2002).

$$
\begin{aligned}
\text{Min} \ & \sum_{j=1}^{n} T_j && && (3.2)\\
\text{s.t.} \ & \sum_{j=1}^{n} z_{ij} = 1 && \forall\, i \in \{1,\dots,n\}, && (3.3)\\
& \sum_{i=1}^{n} z_{ij} = 1 && \forall\, j \in \{1,\dots,n\}, && (3.4)\\
& b_{irj} \geq z_{ij} + z_{r,j+1} - 1 && \forall\, i,r \in \{1,\dots,n\}, i \neq r, j \in \{1,\dots,n-1\}, && (3.46)\\
& b_{irj} \leq z_{r,j+1} && \forall\, i,r \in \{1,\dots,n\}, i \neq r, j \in \{1,\dots,n-1\}, && (3.47)\\
& b_{irj} \leq z_{ij} && \forall\, i,r \in \{1,\dots,n\}, i \neq r, j \in \{1,\dots,n\}, && (3.48)\\
& b_{irj} \geq 0 && \forall\, i,r \in \{1,\dots,n\}, i \neq r, j \in \{1,\dots,n\}, && (3.49)\\
& S_1 \geq 0, && && (3.50)\\
& S_j \geq S_{j-1} + \sum_{i,r=1; i\neq r}^{n} b_{ir,j-1} w_{ir} && \forall\, j \in \{2,\dots,n\}, && (3.51)\\
& C_j = S_j + \sum_{i=1}^{n} z_{ij} p_i && \forall\, j \in \{1,\dots,n\}, && (3.52)\\
& T_j \geq C_j - \sum_{i=1}^{n} z_{ij} d_i && \forall\, j \in \{1,\dots,n\}, && (3.53)\\
& T_j \geq 0 && \forall\, j \in \{1,\dots,n\}, && (3.11)\\
& z_{ij} \in \{0,1\} && \forall\, i,j \in \{1,\dots,n\}. && (3.12)
\end{aligned}
$$

Variable b_{irj} is equal to 1 when job i of sequence position j immediately precedes job r, that is, job r is in position $j+1$ of sequence. This variable is continuous and constraints 3.46, 3.47, 3.48 and 3.49 guarantees that their values are either 0 or 1, so the integrality of the problem is not compromised (NOGUEIRA et al., 2019). Constraint 3.50 set the start time of the first job of the sequence to zero and constraint set 3.51 defines the start time of the other jobs based on equation 3.43 and variable b_{irj}. The completion time of the job at the last machine and its tardiness is given, respectively, by the constraint sets 3.52 and 3.53. The other constraints are the same as presented in the section of permutation flow shop problem.

The proposed formulation is independent of the number of machines, that is, constraints do not depend on m, which simplifies the formulation resolution and size. Despite the higher complexity size that the variables b_{irj} bring to the positional-based model, Stafford and Tseng (2002) showed that this formulation has a competitive execution time for the solution of an NWFSP-SDST problem. They compared it to a precedence-based model. Similar results are also demonstrated for single machine formulations (NOGUEIRA et al., 2019).

Two models for the NWFSP were presented in this section based on its representation as a traveling salesman problem: a precedence-based formulation for the $F_m|nwt, s_{irk}|C_{max}$, and a positional-based formulation for the $F_m|nwt, s_{irk}|\sum T_j$. In the next chapter, both will be extended for the bi-objective HT scheduling problem to minimize the production line's total energy costs and total tardiness. Despite the mentioned strengths of the positional formulations, a comparative study will be held for the practical scheduling case.

The following section formally defines a multi-objective optimization problem and explores existing solving approaches for multi-objective no-wait flow shop problems. It clarifies how the integration of the presented models and mathematical programming can support the steel industry.

3.2 Multi-objective No-Wait Flow Shop Problems

A multi-objective optimization problem (MOP) can be generally described as:

$$\text{Min } \boldsymbol{f(x)} = \{f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_o(\boldsymbol{x})\}, \tag{3.54}$$

$$\text{s.t. } \boldsymbol{g(x)} = \{g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \ldots, g_p(\boldsymbol{x})\} \leq 0, \tag{3.55}$$

$$\boldsymbol{h(x)} = \{h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_q(\boldsymbol{x})\} = 0, \tag{3.56}$$

where $\boldsymbol{x}$ is the set of decision variables, belonging to the feasible region X, which is a subset of the search space; o is the number (≥ 2) of objective functions; whereas p and q are respectively the numbers of inequality constraints and equality constraints functions (LEI, D., 2008).

The notion of optimality in multi-objective problems is different from that seen in single-objective problems, since it is not possible to find a single alternative solution $\boldsymbol{x}^* = (x_1^*, x_2^*, \ldots, x_o^*)^T$ that optimizes simultaneously all existing objectives, which are now naturally in conflict condition. To enable a strict comparison of solutions in a MOP and to identify which are the optimal solutions in solving these problems, it is necessary to define the concept of dominance:

Definition 1 - Dominance Criterion - given two solutions $\boldsymbol{x^1}$ and $\boldsymbol{x^2}$, belonging to the set of viable solutions X, it is said that $\boldsymbol{x^1}$ dominates $\boldsymbol{x^2}$ ($\boldsymbol{x^1} \prec \boldsymbol{x^2}$) in the space of objectives if, and only if:

- $\forall i \in \{1, 2, \ldots, o\}, f_i(\boldsymbol{x^1} \leq \boldsymbol{x^2}$ - $\boldsymbol{x^1}$ *is not worse than* $\boldsymbol{x^2}$ *in any of the objectives;*
- $\exists i \in \{1, 2, \ldots, o\} \mid f_i(\boldsymbol{x^1} < \boldsymbol{x^2}$ - *in at least one of the objectives* $\boldsymbol{x^1}$ *is better than* $\boldsymbol{x^2}$*.*

The dominance criterion presented allows partial ordering in the objective space, which means that some solutions can be compared, while others are incomparable. Solutions that are not dominated by any other are called Pareto-optimal solutions (or efficient solutions) and are strictly defined as:

Definition 2 - Pareto-optimal solution - a $\boldsymbol{x}^*$ *solution is said to be Pareto-optimal if, and only if:*

- $\nexists \boldsymbol{x} \in X \mid \boldsymbol{x} \prec \boldsymbol{x}^*$ - *if there is no other feasible point that dominates it.*

The definitions made lead to an immediate consequence: it is impossible to seek improvement in any of the objectives of a Pareto-optimal solution without deteriorating at least one of the other objectives. Therefore, the entire set of efficient solutions in the decision variables space (known as the Pareto-optimal set) are solutions to the multi-objective optimization problem. When viewed in the objective space, this set is called the Pareto front, since the solutions are distributed only in part of the ends of the feasible region.

Following is a literature review of MOP solving approaches that support the methodology proposed in this work.

3.2.1 Solving approaches

An extensive survey on no-wait flow shop scheduling problems can be found in Ali Allahverdi (2016). From this review, it is possible to observe that multi-objective NWFSP (MONWFSP) studies appeared in the literature at the beginning of the 21st century and are much less extensive than single-objective ones. The comprehensive review also shows that metaheuristics (either local search or population-based ones) are the preferred solving methods.

Table 3.1 shows examples of recent papers that solve the MONWFSP problem. The problems are described by the triplet $\alpha|\beta|\gamma$, such as stated in Graham et al. (1979). In this notation, the greek letters represent, respectively, the scheduling problem machine environment, processing characteristics, and optimality criteria. The F stands for flow shop. The β field can have multiple entries to show different aspects of the process, such as the no-wait restrictions (nwt). The γ field specifies the scheduling objectives, which can be one or more. Examples of the objectives of Table 3.1 are the total completion time (C_{max} - makespan), total tardiness (TT), total weighted tardiness ($\sum w_j T_j$), and total energy costs (TEC).

Table 3.1 – Recent papers (last three years) on MONWFSP.

Reference	Problem	Methodology (comments)
Jingjing Wang et al. (2017)	$F\|nwt\|C_{max}, TEC$	Cooperative algorithm that combines random and NEH population initialization, and local search intensifications.
Lin, Lu, and Ying (2018)	$F\|nwt\|C_{max}, \sum w_j T_j$	Cloud theory-based Iterated Greedy algorithm with initialization through NEH-based heuristic.
Allahverdi, Aydilek, and Aydilek (2018)	$F\|nwt\|C_{max}, TT$	Combination of Simulated Annealing and insertion algorithm with EDD (Earliest Due Date) initialization.
Yeh and Chiang (2018)	$F\|nwt\|C_{max}, TT$	MOEA/D with problem specific crossover operations and population initialization through EDD and combination of heuristics.
Wu and Che (2019)	$F\|nwt\|C_{max}, TEC$	Multi-objective Variable Neighborhood Search with speed scaling heuristic and initial solutions generated by problem specific heuristics.

Table 3.1 suggests that most recent approaches combine different heuristics and metaheuristics for the solution of the MONWFSP. However, these hybrid methods do not consider using mathematical programming, which is the strategy explored in this work.

Some recent works have already applied matheuristic methodologies for solving single-objective problems in the scope of production scheduling. Fonseca, Nogueira, and Ravetti (2019) recommends a Lagrangian relaxation technique combined with heuristics to solve a cross-docking scheduling problem, reaching tight bounds for small and large instances. M'Hallah (2014) proposes a hybrid Variable Neighborhood Search and Mixed Integer Programming to solve difficult instances of a m-stage permutation flow shop. Ta, Billaut, and Bouquard (2018) use a mathematical heuristic to minimize the total tardiness of a m-machine flow shop scheduling problem. It integrates a neighborhood-based algorithm with iterative executions of a MILP solver to re-optimize small windows of jobs (fix-and-optimize approach) in the metaheuristic course. Lin and Ying (2016), in the solution of an NWFSP to minimize the makespan, use a constructive heuristic for the generation of an initial solution that is later simplified, modeled, and solved as a Traveling Salesman Problem to optimality. Mönch and Roob (2018) design a matheuristic framework to solve a parallel batch machine scheduling problem with incompatible job families. First, using MP to solve a batch formation problem and, later, running a population-based heuristic that will schedule the batches in the machines to minimize the total weighted tardiness. The successful approaches listed above suggest that matheuristic can take advantage of exact method/metaheuristic different search properties for solving complex problems and large instances, making it promising for the solution of real scheduling applications.

Different metaheuristics have been recently applied to solve multi-objective scheduling problems. Some examples are genetic algorithms (ZHENG et al., 2019; ANJANA; SRIDHARAN; RAM KUMAR, 2019), and multi-objective estimation of distribution algorithms (SHAO; PI; SHAO, 2019; LIU; YANG, et al., 2020). The metaheuristic employed in this research is a multi-objective version of the Variable Neighborhood Search (VNS) algorithm presented by Mladenović and Hansen (1997). This method relies on systematically changing neighborhoods while searching for solutions in the exploration space. Its efficiency is based on three factors: the local minimum of a neighborhood is not necessarily the local minimum of another; the global minimum is a local minimum for all neighborhoods; in several problems, the local minimum between neighborhoods is relatively close (HANSEN et al., 2010).

MOVNS has shown good results in the solution of scheduling problems since its proposal by Geiger (2004) to solve the permutation FSP. It also appears in recent researches on multi-objective no-wait flow shop (WU; CHE, 2019), multi-objective two agent flow shop (LEI, Deming, 2015), multi-objective hybrid flow shop (JIANG et al., 2015), multi-objective hybrid flow shop with sequence-dependent setup times (TIAN; LI; LIU, 2016; SIQUEIRA; SOUZA; SOUZA, 2018) and multi-objective blocking flow shops (WANG, F. et al., 2018). Results from these manuscripts suggest that MOVNS can outperform multi-objective state-of-art algorithms Non-dominated Sorting Genetic Algorithm II (NSGA-II) and Strength Pareto Evolutionary Algorithm (SPEA2) in FSP related applications. However, there are no researches that combine this multi-objective metaheuristic with mathematical programming, as will be presented next. In terms of the energy consumption, the mechanism applied in this work has lead to optimal energy solutions, which is appointed by Wu and Che (2019) and Fucai Wang et al. (2018) as an important investigation point for industries.

3.3 Matheuristics

Matheuristics, or mathematical heuristics, are hybrid algorithms that combine heuristics and metaheuristics (MH) with mathematical programming (MP) for the solution of real and complex optimization problems. Heuristics are solving methods that employ simple rules to find good enough solutions for a specific problem, based on its prior observation and experience. Metaheuristics are high-level strategies that guide the search on non-specific problems, although they can also rely on heuristics knowledge, to find near-optimal solutions within a reasonable time (RAIDL; PUCHINGER, 2008). They are separated into local search and population-based methods (BLUM; ROLI, 2003). Mathematical programming is a branch of operations research that relies on mathematical formulations and techniques to find optimal solutions in decision-making tasks. These methods ensure that the solutions found are optimal, but are restricted to small instances in NP-Hard problems (RAIDL; PUCHINGER, 2008). Matheuristics lie on the idea of exploring the strength of these optimization methods concerning the quality of solutions and the search time used to find them (CROCE; GROSSO; SALASSA, 2014; TALBI, 2016).

Puchinger and Raidl (2005), in one of the first reviews of matheuristics for combinatorial optimization problems, highlight the combined use of local search and population algorithms with integer and linear programming methods (e.g., Branch and Bound, Branch and Cut, Dynamic Programming, and Lagrangian relaxation). They show examples in the solution of scheduling, assignment, routing, and cutting problems, widely applied in industrial fields, and known for their NP-hard complexity. They also propose a taxonomy for this recent research field that separates the combined methods into collaborative or integrative, as exploited in Figure 3.4.

In the collaborative combinations, the algorithms exchange information among themselves but are not part of each other. They can be executed sequentially, parallel,

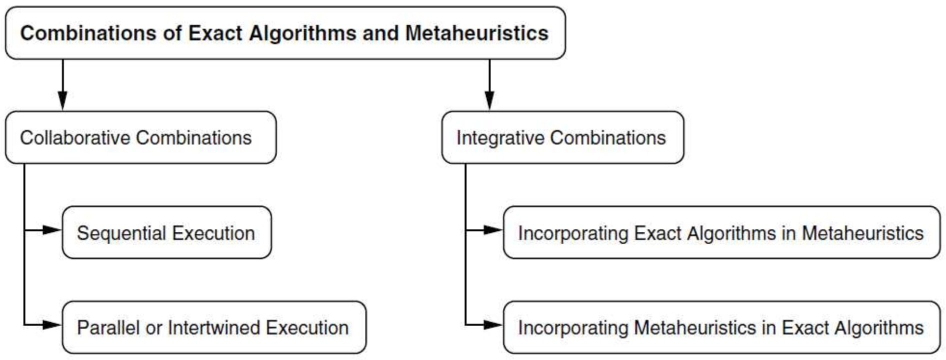

Combinations of Exact Algorithms and Metaheuristics
Collaborative Combinations
Sequential Execution
Parallel or Intertwined Execution
Integrative Combinations
Incorporating Exact Algorithms in Metaheuristics
Incorporating Metaheuristics in Exact Algorithms

Figure 3.4 – Classification of matheuristics.

Proposed by Puchinger and Raidl (2005).

or intertwined. The opposite occurs in the integrated combinations, where there is a main algorithm, which can be an exact method or a metaheuristic, and at least one slave algorithm, which optimizes subproblems. Below are some examples of approaches in these four sub-classifications.

- Sequential Execution: normally evolves two phases, in which the first is executed by the exact method and the second by the metaheuristic, or vice-versa. In some cases, the first algorithm is a preprocessor, and it is used to generate initial solutions or bounds for the second algorithm. There are other cases in which the second method is a postprocessor that improves the already found solutions. Examples of these approaches are: Feltl and Raidl (2004) solves a general assignment problem with the initialization of the Genetic Algorithm population through Linear Programming Relaxation (LP-Relaxation), and Applegate et al. (1998) runs multiple Iterated Local Search algorithms to find a set of solutions for a traveling salesman problem later merged into a restricted graph and solved to optimality. In some cases more than two phases are presented, such as in Lin and Ying (2016) that alternates executions between two constructive heuristics and a relaxed Mixed Integer Programming model to solve the makespan minimization of a no-wait flow shop.

- Parallel or Intertwined Execution: this topology is the less frequent combination of algorithms. Puchinger and Raidl (2005) cites two frameworks for the solution of combinatorial problems. One of them is a job shop scheduling problem, in which multiple agents, from different classes of algorithms, are called asynchronously to exchange information or to alter a set of solutions present in shared memory.

- Incorporating Exact Algorithms in Metaheuristics: mathematical programming is integrated into a metaheuristic and can be used, for instance, to explore large neighborhoods in local search algorithms and as decoders in support of evolutionary algorithms operations. Examples of the first approach are demonstrated by Haouari and Ladhari (2003), that introduces a new neighborhood in a local search algorithm, based on a Branch and Bound tree search, for the solution of a permutation flow shop, and also by Fleszar, Charalambous, and Hindi (2012), that uses Mixed Integer Programming in a Variable Neighborhood Descent large search procedure for the solution of parallel machine scheduling. In the evolutionary field, Jahuira (2002) solves a traveling salesman problem with a Genetic Algorithm whose crossover operations are performed by a Branch and Bound algorithm and by a Minimum Spanning Tree heuristic.

- Incorporating Metaheuristics in Exact Algorithms: the main algorithm is an exact method embedded with metaheuristics. MH usually supports the selection of nodes, generation of upper bounds, and cutting planes in Branch and Bound methods (JOURDAN; BASSEUR; TALBI, 2009). An example of these approaches is presented by Kostikas and Fragakis (2004) as a nested Genetic Programming algorithm that improves the node selection of a general Branch and Bound algorithm. In this class, there are also MP algorithms that apply the spirit of local search metaheuristics, although they are not hybrid. These methodologies are known as MILP based heuristics, and two renowned procedures of this class are Local Branching, from Fischetti and Lodi (2003), and Relaxation Induced Neighborhood Search (RINS), proposed by Danna, Rothberg, and Pape (2005).

The ideas of Puchinger and Raidl (2005) were expanded in papers from Jourdan, Basseur, and Talbi (2009) and Talbi (2016), that propose a different classification for the hybridization of MH and MP. Other reviews of matheuristics can also be found in Dumitrescu and Stützle (2003) and Blum, Puchinger, et al. (2011). Despite the different taxonomies and emphasis of each paper, Figure 3.5 shows the common sense of information exchanged from one method to another. Usually, the exact methods are favored by the upper bounds, incomplete solutions, and subproblems solved by the metaheuristics in lower computational time. On the other hand, metaheuristics can take advantage of exact methods lower bounds, relaxed optimal solutions, and partial solutions to improve the quality of the local searches and the solutions of population-based algorithms.

Jourdan, Basseur, and Talbi (2009) accentuates the growth of matheuristics from the end of the 20th century to the beginning of the 21st century. However, in their survey few researchers are dealing with this methodology in multi-objective problems. Ehrgott and Gandibleux (2008) also emphasize how marginal are these strategies in multi-objective contexts, despite their importance in many practical problems, which enhance the research possibilities in this subject.

There are different techniques to solve multi-objective problems and to find a set

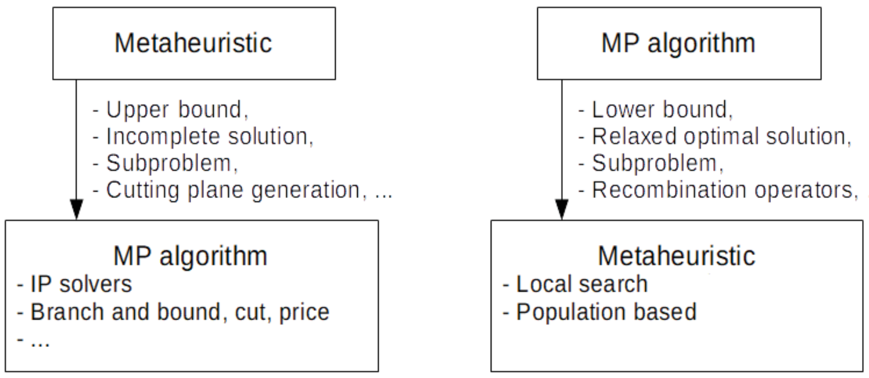


Figure 3.5 – Information exchanged between metaheuristics and exact methods.

of solutions that show the trade-offs between the objectives. The methodology chosen depends on the decision-making process, classified in a priori[1], iterative[2], and a posteriori (SILVA; BURKE; PETROVIC, 2004). In the last one, which will be the focus of this work, the decision-maker analyzes its preferences after optimization. Therefore, the optimization result is a set of solutions with interesting trade-offs among the objectives. These solutions are called non-dominated[3] and their set is known as the Pareto front.

Examples of mathematical heuristics for the a posteriori resolution of multi-objective combinatorial optimization problems are found in Gandibleux and Freville (2000), Basseur et al. (2004), and Przybylski, Gandibleux, and Ehrgott (2010). The first is applied to a $0-1$ knapsack problem and introduces the exact method as a preprocessing algorithm to reduce the search space of a Tabu Search process. In Basseur et al. (2004), a bi-objective flow shop problem is solved through the use of MP and evolutionary algorithms. The exact approach is a Two-Phase Method proposed by Ulungu and Teghem (1995) that is combined with an Adaptive Genetic/Memetic algorithm in three different ways: in a sequential execution (MH for initial solutions and MP for polishing), in a very large search neighborhood, and for the exploration of regions of the Pareto front found by the metaheuristic. The third work also deals with the Two-Phase Method in an assignment problem but expands the procedure to more than two objectives.

In this research, a sequential execution between a MILP solver and a MOVNS algorithm is proposed for the solution of the bi-objective no-wait flow shop problem. The mathematical programming is used to find good initial solutions that are further explored by the local search metaheuristic. A solution polishing procedure, based on a MILP heuristic, is also presented and is used to improve the initial solutions first generated and

[1] The decision-makers express their preferences for each objective before the optimization, and one or various executions are performed to find satisfying solutions.
[2] Decision makers are consulted during the optimization process to adjust the preferences and guide the search through satisfying solutions.
[3] It is said that a solution is dominated in multi-objective optimization if there is another solution that is better than it in at least one objective without being worse in the other ones.

also to improve the Pareto front solutions found by the metaheuristic.

4 Proposed Multi-Objective Matheuristic

4.1 Introduction

The multi-objective matheuristic proposed to solve the HT scheduling problem consists of five main steps. The first step is the generation of two initial solutions through mathematical programming. Following, these solutions go through MILP heuristics for the improvement of the total tardiness and total energy cost objectives. These procedures will be referred to in this text as solution polishing. Then, the two solutions are used to start a Multi-objective General Variable Neighborhood Search algorithm. Finally, some solutions are chosen from the final approximated Pareto to go through final polishing. Figure 4.1 shows the proposed matheuristic flowchart.

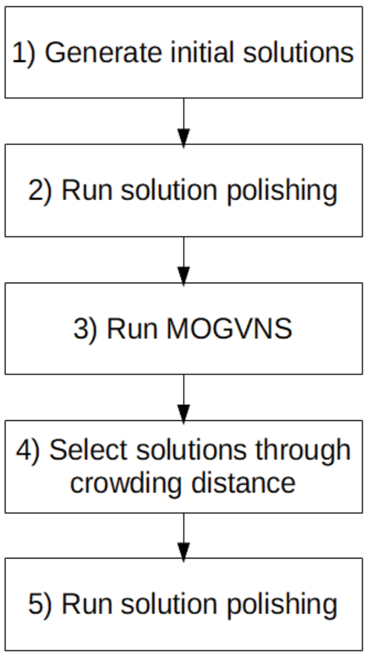

1) Generate initial solutions
2) Run solution polishing
3) Run MOGVNS
4) Select solutions through crowding distance
5) Run solution polishing

Figure 4.1 – Flowchart of proposed matheuristic.

The next sections detail the main steps of the proposed methodology, starting by presenting the bi-objective models for the minimization of the total tardiness and total energy costs of the heat treatment production line.

4.2 HT Scheduling Mathematical Models

The first and second steps of the proposed matheuristic are based on two particular Mixed Integer Linear Programming (MILP) models for the HT scheduling problem. The first model stands on the no-wait flow shop positional-based formulation, presented in Section 3.1.2.2, and the second is found on the TSP formulation introduced in Section 3.1.2.1, being a precedence-based model. The parameters and variables used in both models are summarized in Table 4.1 and Table 4.2.

Table 4.1 – Notation for the heat treatment scheduling parameters.

n	number of jobs.
m	number of machines.
d_i	due date of job i.
p_i	total process time of job i throughout all machines $[h]$.
w_{ir}	waiting time at start of job r, when job i precedes job r $[h]$.
v'_i	total volume of gas used during execution of job i throughout all machines $[Nm^3/h]$.
v''_{ir}	total volume of gas used to setup all machines for job r, when job i precedes job r $[Nm^3/h]$.
g	cost of natural gas per volume $[\$/Nm^3]$.
M	very large number.

Table 4.2 – Notation for the heat treatment scheduling variables.

z_{ij}	positional variable of jobs in sequence. It is equal to 1 when job i is in position j or 0 otherwise.
b_{irj}	precedence variable of the positional model. It is equal to 1 when job i at sequence position j immediately precedes job r, that is, job r is in position $j+1$.
S_j	start time of the job in position j at machine 1.
C_j	completion time of the job in position j at machine m.
T_j	tardiness of the job in position j.
V_j	total volume of gas consumed by the job in position j.
y_{ir}	precedence variable for the jobs in sequence. It is equal to 1 when job i immediately precedes job r or 0 otherwise.
u_i	variable to avoid sub tour in the TSP formulation.
S_i	start time of job i at machine 1.
C_i	completion time of job i at machine m.
T_i	tardiness of job i.

4.2.1 Bi-objective Positional Model

The positional-based model of the quenching and tempering line is based on the classical Wilson (1989) permutation FSP formulation. The job assignment is modeled through binary variables z_{ij}, which is equal to 1 when job i goes into position j of the schedule, otherwise is 0. Index j represents the position of a job in the sequence. Below is the model.

$$\text{Min} \sum_{j=1}^{n} T_j \tag{4.1}$$

$$\text{Min } g\sum_{j=1}^{n} V_j \tag{4.2}$$

$$\text{s.t. } \sum_{j=1}^{n} z_{ij} = 1 \qquad \forall\, i \in \{1,\dots,n\}, \tag{4.3}$$

$$\sum_{i=1}^{n} z_{ij} = 1 \qquad \forall\, j \in \{1,\dots,n\}, \tag{4.4}$$

$$b_{irj} \geq z_{ij} + z_{r,j+1} - 1 \qquad \forall\, i,r \in \{1,\dots,n\}, i \neq r, j \in \{1,\dots,n-1\}, \tag{4.5}$$

$$b_{irj} \leq z_{r,j+1} \qquad \forall\, i,r \in \{1,\dots,n\}, i \neq r, j \in \{1,\dots,n-1\}, \tag{4.6}$$

$$b_{irj} \leq z_{ij} \qquad \forall\, i,r \in \{1,\dots,n\}, i \neq r, j \in \{1,\dots,n\}, \tag{4.7}$$

$$S_1 \geq 0, \tag{4.8}$$

$$S_j \geq S_{j-1} + \sum_{i,r=1;i\neq r}^{n} b_{ir,j-1} w_{ir} \qquad \forall\, j \in \{2,\dots,n\}, \tag{4.9}$$

$$C_j = S_j + \sum_{i=1}^{n} z_{ij} p_i \qquad \forall\, j \in \{1,\dots,n\}, \tag{4.10}$$

$$T_j \geq C_j - \sum_{i=1}^{n} z_{ij} d_i \qquad \forall\, j \in \{1,\dots,n\}, \tag{4.11}$$

$$V_1 \geq \sum_{i=1}^{n} z_{i1} v'_i, \tag{4.12}$$

$$V_j \geq \sum_{r=1}^{n} z_{rj} v'_r + \sum_{i,r=1;i\neq r}^{n} b_{ir,j-1} v''_{ir} \qquad \forall\, j \in \{2,\dots,n\}, \tag{4.13}$$

$$T_j \geq 0 \qquad \forall\, j \in \{1,\dots,n\}, \tag{4.14}$$

$$V_j \geq 0 \qquad \forall\, j \in \{1,\dots,n\}, \tag{4.15}$$

$$b_{irj} \in \{0,1\} \qquad \forall\, i,r \in \{1,\dots,n\}, i \neq r, j \in \{1,\dots,n\}, \tag{4.16}$$

$$z_{ij} \in \{0,1\} \qquad \forall\, i,j \in \{1,\dots,n\}. \tag{4.17}$$

Equations 4.1 and 4.2 are, respectively, the total tardiness (TT) and the total energy costs (TEC) objectives, indexed by the job at sequence position j. The TEC is given by the total volume of gas consumed during the entire sequence execution (sum of V_j), multiplied by the industrial cost of natural gas per volume, g ($\$/Nm^3$). Constraints 4.3 and 4.4 are classic from assignment problems and ensure that job i is assigned to only one position in schedule and that each position j has only one job assigned to it. Variable b_{irj} is equal to 1 when job i of sequence position j immediately precedes job r, that is, job r is in position $j+1$ of sequence. This variable is naturally binary but it can be relaxed since constraints 4.5, 4.6, 4.7, and 4.16 ensure that its value is either 0 or 1, not compromising the integrality of the problem (NOGUEIRA et al., 2019). Constraint 4.8 sets the start time of the first job of the sequence to at least zero and constraint set 4.9 defines the start time of the other jobs. This value is calculated based on the precedence variable b_{irj}. If job i is at position j of sequence and immediately precedes job r ($b_{irj} = 1$), then the start time of job r is at least the start time of job i with the waiting time, w_{ir}, defined in equation 2.3, that will guarantee the no-wait requirement. The completion time of the job at the last machine and its tardiness is given, respectively, by the constraint sets 4.10, 4.11, and 4.14 (see (RONCONI; BIRGIN, 2012)). Constraint 4.12 defines the total volume of NG consumed by the first job in the sequence, that has no setup. The constraint set 4.13 defines the total volume of NG consumed from the second to the last job. In this case, the precedence variable, b_{irj}, is important to determine the gas consumption during idle times (includes setups) while the furnaces are changing their temperatures from job i (position $j-1$) to r (position j). The non-negative value of variable V_j are guaranteed by the constraint set 4.15. Finally, constraint set 4.17 guarantees the binary integrity of the positional variable z_{ij}. This formulation complexity is $O(n^2)$.

4.2.2 Bi-objective Precedence Model

The precedence model of the heat treatment line is based on the no-wait flow shop formulation as an asymmetrical traveling salesman problem. Here, a dummy job is also introduced to identify the start and end of the sequence. Therefore, the number of jobs for this model is given by $n' = n + 1$. Similarly to the makespan model, presented in Section 3.1.2, the total volume of gas consumed during a sequence can be calculated given the precedence variable, y_{ir}, and the volume of gas consumed during production:

$$V_{total} = \sum_{i,r=1}^{n+1} y_{ir}(v'_i + v''_{ir}), \tag{4.18}$$

where, v'_i is the volume of gas consumed to produce job i, and v''_{ir} is the volume of gas consumed during the furnace setup from job i to job r, when job i immediately precedes job r ($y_{ir} = 1$). For the dummy job $n+1$, $v'_{n+1} = 0$ and $v''_{n+1,r} = 0$.

Below is the precedence-based bi-objective model for the HT scheduling problem.

$$\text{Min } \sum_{i=1}^{n'} T_i \tag{4.19}$$

$$\text{Min } g \sum_{i,r=1}^{n'} y_{ir}(v'_i + v''_{ir}) \tag{4.20}$$

$$\text{s.t. } \sum_{r=1}^{n'} y_{ir} = 1 \qquad \forall\, i \in \{1, \dots, n'\}, \tag{4.21}$$

$$\sum_{i=1}^{n'} y_{ir} = 1 \qquad \forall\, r \in \{1, \dots, n'\}, \tag{4.22}$$

$$S_i \geq 0 \qquad \forall\, i \in \{1, \dots, n\}, \tag{4.23}$$

$$S_r \geq S_i + w_{ir} - M(1 - y_{ir}) \qquad \forall\, i, r \in \{1, \dots, n\}, \tag{4.24}$$

$$C_i = S_i + p_i \qquad \forall\, i \in \{1, \dots, n\}, \tag{4.25}$$

$$T_i \geq C_i - d_i \qquad \forall\, i \in \{1, \dots, n\}, \tag{4.26}$$

$$T_i \geq 0 \qquad \forall\, i \in \{1, \dots, n\}, \tag{4.27}$$

$$u_i - u_r + n' y_{ir} \leq n' - 1 \qquad \forall\, i, r \in \{2, \dots, n'\}, \tag{4.28}$$

$$1 \leq u_i \leq n' \qquad \forall\, i \in \{1, \dots, n'\}, \tag{4.29}$$

$$y_{ir} \in \{0, 1\} \qquad \forall\, i, r \in \{1, \dots, n'\}. \tag{4.30}$$

Equations 4.19 and 4.20 are, respectively, the total tardiness (TT) and the total energy costs (TEC) objectives. The total tardiness variable is indexed by the job, and the total energy costs is given by Equation 4.18, multiplied by the industrial cost of natural gas per volume, g ($\$/Nm^3$). Constraint sets 4.21 and 4.22 are the assignment constraints of the ATSP formulation presented on Section 3.1.2.1. Constraint 4.23 sets the start time of all the jobs to at least zero and constraints 4.24 guarantees that, if job i immediately precedes job r, given a very large number, M, the start of job r is at least the start of job i added by the waiting time, w_{ir}. The completion time of the job at the last machine and its tardiness is given, respectively, by the constraint sets 4.25, 4.26, and 4.27. Constraints 4.28 and 4.29 are the subtour elimination constrains from the ATSP formulation. Finally, constraint set 4.30 guarantees the binary integrity of the precedence variable y_{ir}. This formulation complexity is $O(n')$.

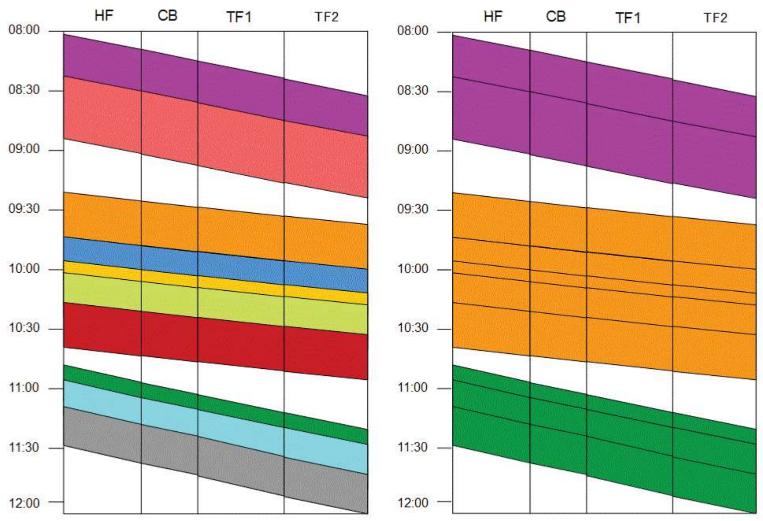
4.3 Generation of Initial Solutions

The first step of the proposed matheuristic is the generation of two initial solutions by exact resolution. The goal is to find good solutions in terms of total tardiness and total energy cost, which can map the Pareto front extremes and support the multi-objective local search algorithm. The bi-objective models presented are split into two single-objective models to generate the two initial solutions that minimize TT and TEC.

They are constructed by removing the objective function, constraints, and variables that are not related to the minimization goal.

These problems can be solved exactly with the use of a MILP solver. However, the steel industry scheduling instances are too large (minimum of 30 jobs) to be determined within a reasonable time. Thus, a grouping strategy of the jobs, that reduces the problem resolution size, is suggested.

Jobs in the QT line that have the same process settings, regardless of their due dates, can be processed together without the need for any machine setup or idle time. They are considered as being from the same job family. As a consequence, in scheduling terms, they can be considered a single job. Figure 4.2 shows an example of how ten jobs can be grouped as three jobs in this steel industry real case. The schedule view is the same as in Figure 2.6. Each machine appears on the horizontal axis, with the width proportional to its capacity, and the time is on the vertical axis. In Figure 4.2a each production lot appears in a different color. In Figure 4.2b, the colors identify the groups of jobs that can be processed together.



(a) Schedule view of ten jobs. (b) Schedule view of three jobs.

Figure 4.2 – Example of ten jobs scheduled in the QT line considered as three.

The jobs are grouped according to their process characteristics. In the x-axis are the machines and in the y-axis the time.

To illustrate the grouping strategy, a fictitious instance of 10 jobs is presented in Table 4.3. The process settings listed per job i are the number of steel pieces to be produced, y_i, the walking beam cycle time, c_i, given in seconds, the furnaces temperatures, t_{ik}, per machine k, and the due date, d_i.

The jobs are grouped by furnace temperatures and cycle time to solve the minimization of TEC. Considering the fictitious instance presented in Table 4.3, the following jobs would be grouped: 1-9, 2-3-8-10, 4, and 5-6-7. Therefore, the minimization of TEC runs

Table 4.3 – Fictitious instance with 10 jobs for demonstration of the procedures based on the job families.

i	y_i	$c_i[s]$	$t_{i1}[°C]$	$t_{i2}[°C]$	$t_{i3}[°C]$	$t_{i4}[°C]$	d_i
1	45	40	825	0	300	400	1
2	90	20	850	0	325	475	1
3	70	20	850	0	325	475	2
4	60	30	850	0	325	475	2
5	30	30	815	0	325	475	2
6	50	30	815	0	325	475	2
7	80	30	815	0	325	475	2
8	25	20	850	0	325	475	2
9	85	40	825	0	300	400	1
10	90	20	850	0	325	475	2

for an instance of four jobs. In this case, the new group of jobs would have, respectively, 130, 275, 60, and 160 steel pieces to be produced. The due date is used to sequence the group's jobs, following an Earliest Due Date (EDD) approach.

For the TT minimization, the jobs are grouped by due dates, temperatures, and cycle time. Jobs 1 and 2 of the fictitious instance have the same due date; however, they cannot be grouped, because they have different process settings. Therefore, the groups would be: 1-9, 2, 3-8-10, 4, and 5-6-7, with, respectively, 130, 90, 185, 60, and 160 steel pieces to be produced. Then, the minimization of TT would proceed with five jobs.

The total energy costs of a schedule in the HT line depends on the volume of natural gas consumed during the steel pieces manufacture (equation 2.5) and the amount of gas consumed during the furnaces idle times (equation 2.6). In the proposed simplification, the volume of gas consumed by a group of jobs during the steel pieces production results from its processing times sum, which is constant and independent of the order of jobs inside the group. Therefore, the minimization of TEC will only be affected by the sequence of the groups. Consequently, if there is an optimal solution for the minimization of TEC in the reduced scenario, this solution is optimal for the original problem. Thus, a Pareto front extreme value for the total energy costs can be known, provided that the reduced problem can be solved. The latter postulate does not apply for the total tardiness minimization problem. In this context, the order of the jobs inside the groups interferes directly in the individual job delays, so there is no guarantee that an optimal solution found for the reduced problem is optimal for the original one. Despite that, it is expected that a good enough TT solution can be found, helping to map the Pareto front extreme for total tardiness.

The next section explains a solution polishing method, also based on grouping the jobs by family, to improve a solution in terms of total tardiness and total energy costs. It is applied over the initial solutions generated to bring them as close as possible to the

Pareto front extremes of the problems.

4.4 MILP Heuristics for Solution Polishing

In this work, we define two fix-and-optimize approaches to improve a solution total tardiness and total energy costs. In the fix-and-optimize strategy, a significant part of the variables is fixed, with the expectation that the resolution of the model considering only the remaining ones can lead to an improved solution within a reasonable time. The number of fixed variables is set in such a way that the resultant model can be solved exactly into a given target time.

The MILP heuristics for the solution polishing are based on grouping the jobs by temperature, cycle time, and due date (similar grouping strategy to minimize TT). Each group will be referred to as a family. To exemplify this concept, Table 4.4 shows the 10 jobs of Table 4.3 and the families they belong, also identified by numbers.

Table 4.4 – Demonstration of the jobs families for fictitious instance of Table 4.3

jobs	1	2	3	4	5	6	7	8	9	10
families	1	2	3	4	5	5	5	3	1	3

When two jobs from the same family immediately precede each other, they can be processed together, as shown in the previous section. Each group of jobs with the same process characteristics, and due date, will be referred to as a batch. Table 4.5 illustrates a possible sequence for the jobs of Table 4.4 with its correspondent batches. The batches are defined based on the families.

Table 4.5 – Demonstration of the batches for a random sequence of jobs

sequence	2	3	8	10	4	1	9	5	6	7
batches	1	2	2	2	3	4	4	5	5	5

The total tardiness of the sequence is minimized if the TT of a batch is minimized. Therefore, using the positional-based model and a fix-and-optimize approach, it is possible to iterate over the sequence batches to try to improve the overall solution total tardiness. At each iteration, a batch is selected, and the position of the jobs outside the group is fixed. With that, the MILP solver works on a smaller problem. Also, since the order of the jobs inside a batch does not affect the sequence total energy costs, a single objective model, with only the variables and constraints related to TT, can be run.

Differently, from the total energy costs minimization perspective, only the order of the batches is relevant. Therefore, a fix-and-optimize approach is applied based on the HT precedence model. The precedence of the jobs inside the batches are kept fixed and the MILP solver can only change the precedence relation between the batches to try to further minimize the TEC. In this case, the total tardiness of the sequence can be affected by the TEC minimization. Therefore, the precedence bi-objective model is modified to a single-objective model, and the total tardiness objective becomes the constraint shown in equation 4.31, in which TT_{max} is the total tardiness of the original solution.

$$\sum_{i=1}^{n'} T_i \leq TT_{max}. \tag{4.31}$$

Both fix-and-optimize approaches reduce the problem resolution size. However, there may be cases in which the number of jobs inside a batch is too large to minimize the TT, or that there are too many batches, being difficult to run the solver for the minimization of TEC. For this scalability issue, a complementary heuristic based on the AXB framework proposed from Ta, Billaut, and Bouquard (2018) is used.

Figure 4.3 illustrates how the AXB framework works. At first, a sequence S is divided into three parts: A, X, and B. The A and B parts are fixed, and the X are the variables of the sequence that will be re-optimized. The X set has a fixed size h and goes from sequence position r to $r+h-1$. Re-optimization of X leads to X' and a new sequence $S' = AX'B$. The AXB framework is iterative, and the X window moves from position 1 of the sequence up to $n-h$ position, one position at a time if there is no improvement in the objective function of S. If an improvement appears, the re-optimization window moves h positions, and the next re-optimization starts from position $r+h-1$.

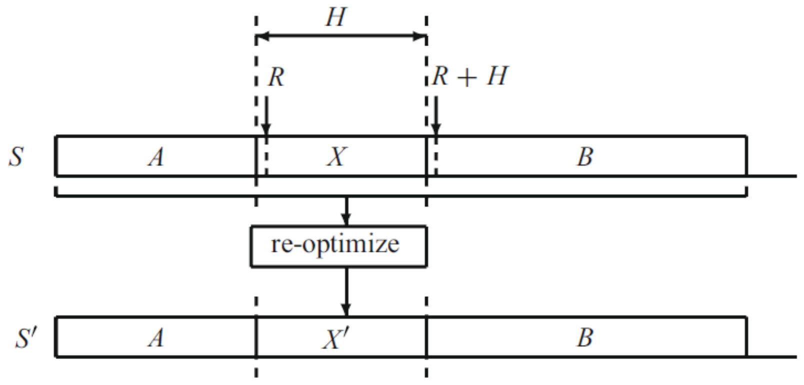


Figure 4.3 – Illustration of proposed fix-and-optimize algorithm.
Based on the AXB framework from Ta, Billaut, and Bouquard (2018).

Based on the AXB framework concept, Algorithm 4.1 details the complete steps of the solution polishing for the total tardiness objective. The parameters of the algorithm are a sequence S, a vector $G = \{G_1, G_2, \ldots\}$, where $G_f = \{g_1, g_2\}$ indicates, respectively, the start index and size of a batch f, the total number of jobs n, the maximum size of the fixed windows h and the maximum solver execution time for the re-optimization, t_{max}. The procedure runs until all the batches are re-optimized, and batches with only one job are ignored (line 5). Given a batch f, the re-optimization window starts, r, will be equal to the batch starting index. The window size will be the smaller value between the batch size and the user-defined parameter h. If the batch size is higher than parameter h a re-optimization loop goes on, as described in the AXB framework, from the beginning of the batch until its end, replacing sequence S whenever its total tardiness is reduced.

Algorithm 4.1 Pseudocode of the solution polishing for the total tardiness objective.

```
1: function SOLUTION-POLISHING-TT($S, G, n, h, t_{max}$)
2:     $f \leftarrow 1$
3:     repeat
4:         $\{r_b, h_b\} \leftarrow G_f$
5:         if $h_b > 1$ then
6:             $r \leftarrow r_b$
7:             $h \leftarrow$ MIN($h_b, h$)
8:             repeat
9:                 $S' \leftarrow$ FIX-AND-OPTIMIZE($S, n, r, h, t_{max}$)
10:                if $TT(S') < TT(S)$ then
11:                    $S \leftarrow S'$
12:                    if $r + h \leq r_b + h_b - h$ then
13:                        $r = r + h - 1$
14:                    end if
15:                end if
16:                $r = r + 1$
17:            until $r > r_b + h_b - h$
18:        end if
19:        $f = f + 1$
20:    until $f > |G|$
21:    return $S$
22: end function
```

Figure 4.4 illustrates how the total tardiness solution polishing works for a sequence of ten jobs and a maximum re-optimization window of 2 jobs. The red rectangle highlights the batch under re-optimization, and the gray squares are the fixed sequence positions.

Algorithm 4.2 details the complete steps of the solution polishing for the total energy costs. The parameters are the same from the TT procedure, but there are not iterations over batches. The batches are considered as single jobs, and the AXB framework is applied straight over the batches.

Figure 4.5 illustrates how the total energy costs solution polishing works for a sequence of ten jobs and a maximum re-optimization window of 2 jobs. The red rectangles highlight the group of batches that are being re-optimized, and the gray squares represent the pair of jobs that have their precedence fixed at each iteration.

This solution polishing is applied in the two initial solutions generated by mathematical programming and also in the approximated Pareto solutions found by the multi-objective local search metaheuristic described in the next section. First, the TT

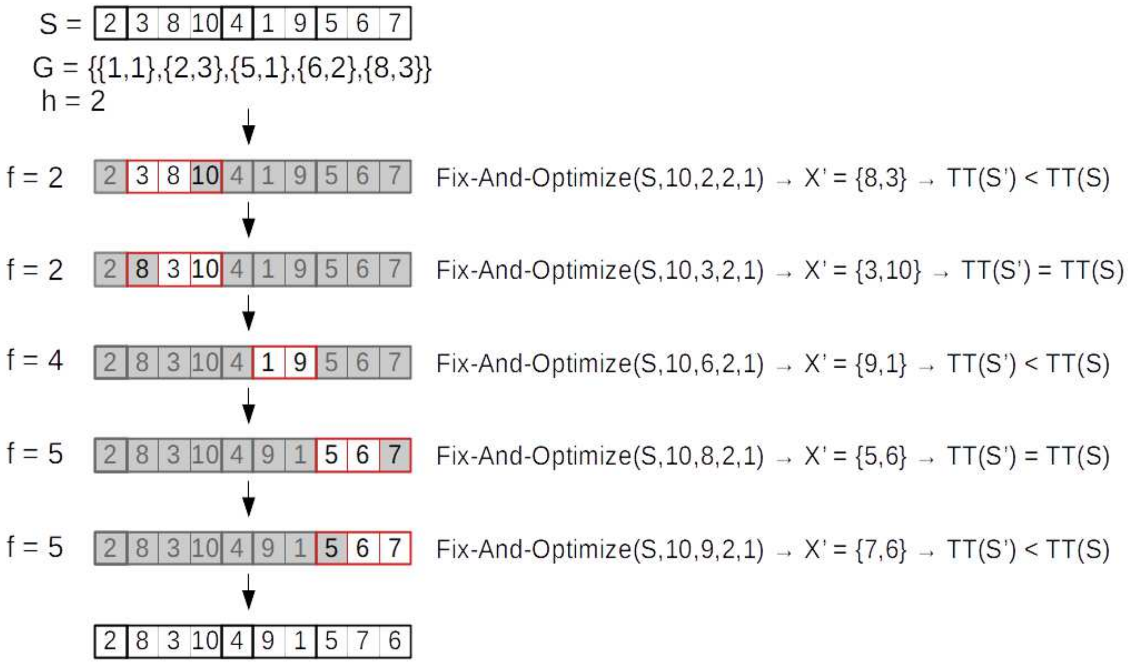

S = 2 3 8 10 4 1 9 5 6 7
G = {{1,1},{2,3},{5,1},{6,2},{8,3}}
h = 2
f = 2
Fix-And-Optimize(S,10,2,2,1) → X' = {8,3} → TT(S') < TT(S)
f = 2
Fix-And-Optimize(S,10,3,2,1) → X' = {3,10} → TT(S') = TT(S)
f = 4
Fix-And-Optimize(S,10,6,2,1) → X' = {9,1} → TT(S') < TT(S)
f = 5
Fix-And-Optimize(S,10,8,2,1) → X' = {5,6} → TT(S') = TT(S)
f = 5
Fix-And-Optimize(S,10,9,2,1) → X' = {7,6} → TT(S') < TT(S)
2 8 3 10 4 9 1 5 7 6

Figure 4.4 – Demonstration of TT solution polishing.

Algorithm 4.2 Pseudocode of the solution polishing for the total energy cost objective.

```
 1: function SOLUTION-POLISHING-TEC($S, G, n, h, t_{max}$)
 2:     if $|G| > 1$ then
 3:         $h \leftarrow$ MIN($|G|, h$)
 4:         $f \leftarrow 1$
 5:         repeat
 6:             $r_b \leftarrow G_f[0]$
 7:             $h_b \leftarrow \sum_{p=f}^{f+h-1} G_p[1]$
 8:             $S' \leftarrow$ FIX-AND-OPTIMIZE($S, n, r_b, h_b, t_{max}$)
 9:             if $TEC(S') < TEC(S)$ then
10:                 $S \leftarrow S'$
11:                 if $f + h \leq |G| - h + 1$ then
12:                     $f = f + h - 1$
13:                 end if
14:             end if
15:             $f = f + 1$
16:         until $f > |G| - h + 1$
17:     end if
18:     return $S$
19: end function
```

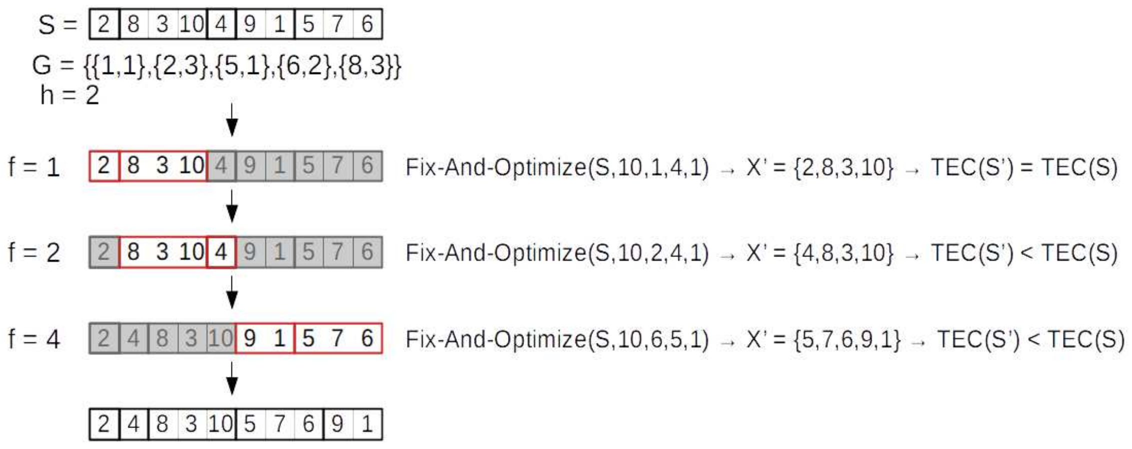

S = 2 8 3 10 4 9 1 5 7 6
G = {{1,1},{2,3},{5,1},{6,2},{8,3}}
h = 2
f = 1
Fix-And-Optimize(S,10,1,4,1) → X' = {2,8,3,10} → TEC(S') = TEC(S)
f = 2
Fix-And-Optimize(S,10,2,4,1) → X' = {4,8,3,10} → TEC(S') < TEC(S)
f = 4
Fix-And-Optimize(S,10,6,5,1) → X' = {5,7,6,9,1} → TEC(S') < TEC(S)
2 4 8 3 10 5 7 6 9 1

Figure 4.5 – Demonstration of TEC solution polishing.

polishing is applied, then the TEC.

4.5 Multi-objective Metaheuristic

The metaheuristic employed in this research is based on the Multi-objective General Variable Neighborhood Search (MOGVNS) algorithm proposed by Duarte et al. (2015).

The MOGVNS is split into three consecutive steps: (i) shaking of the solution to get out of possible local valleys; (ii) variable neighborhood descent search for solution refinement; (iii) neighborhood change until the reach of the stop criterion. Algorithm 4.3 shows the flow between these steps.

The main difference between the MOGVNS and its single-objective version lies in the solution concept. In the mono-objective algorithm, we focus on finding the best possible solution, a single point/vector. In the multi-objective version, our focus is on finding a set of efficient points (non-dominated solutions), represented in Algorithm 4.3 by the parameter E. The other parameters of the procedure are the neighborhood size for shaking moves (q_{max}), the neighborhood size for descent search (l_{max}), and the maximum execution time (t_{max}), which will be the stop criterion.

Given the initial set of non-dominated solutions E, generated with the use of mathematical programming, the q_{max} shaking neighborhoods are visited repeatedly until the stop criterion is reached. A new set of solutions E' is generated from the q shaking neighborhood of set E. This set goes later for a search with l_{max} descent neighborhoods. Finally, the set E'' generated by the descent method is compared to the original set E, and the current shaking neighborhood q is changed. The initial set E is updated with the non-dominated solutions found, and the loop starts all over again.

A solution x is represented by a permutation of jobs, and four neighborhoods are

Algorithm 4.3 Pseudocode of MOGVNS.

```
1: function MOGVNS-FI($E, q_{max}, l_{max}, t_{max}$)
2:     repeat
3:         $q \leftarrow 1$
4:         repeat
5:             $E' \leftarrow$ MOShake($E, q$)
6:             $E'' \leftarrow$ MOVND-FI($E', l_{max}$)
7:             $E, q \leftarrow$ MONeighborhoodChange($E, E'', q$)
8:         until $q > q_{max}$
9:         $t \leftarrow$ CpuTime()
10:    until $t > t_{max}$
11:    return $E$
12: end function
```

considered for MOGVNS. The first two are standard neighborhoods for machine scheduling problems. They represent a set of permutation operations over the n jobs of the solution x. The first one (N_1) consists of exchanging the position of two jobs i and r of the sequence and the second (N_2) on removing job i from its position and inserting it at the position of job r. The cardinalities of the neighborhoods are, respectively, $n(n-1)/2$ and $(n-1)^2$. The next two neighborhoods, N_3 and N_4, consist, respectively, of the same movements, but they are performed over the batches (the jobs from the same batch cannot be moved separately). This batching approach appears on papers from Schaller (2012) and Shen, Gupta, and Buscher (2014) to solve single-objective flow shop problems to minimize, respectively, the total tardiness and makespan. Figure 4.6 exemplifies the movements from the four neighborhoods. All neighborhood movements lead to viable solutions.

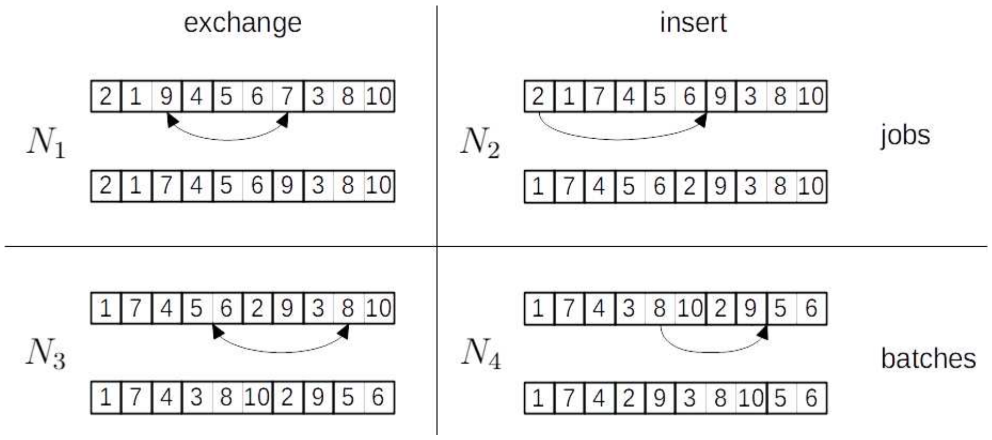


Figure 4.6 – Example of the four neighborhoods.

The multi-objective version of the shaking procedure, MOShake, is presented in Algorithm 4.4. Its primary goal is to perturb the set of efficient points to promote searches on new spaces. The Shake function is responsible for disturbing each solution of set E, according to neighborhood q, to create a new set of points E', that, in this case, may be dominated. The shaking strategy to create a new x' solution can vary from problem to problem. In this research, a solution is perturbed by q random movements of the sequence. Each procedure is an exchange, or an insert operation over the batches, which are chosen at random (considering a uniform probability distribution). This random selection of both neighborhoods and switching jobs benefits the search space exploration to escape from local-optimal valleys.

Algorithm 4.4 Pseudocode of multi-objective shake.

```
1: function MOShake($E, q$)
2:     $E' \leftarrow \emptyset$
3:     for all $x \in E$ do
4:         $x' \leftarrow$ Shake(x,q)
5:         $E' \leftarrow E' \cup \{x'\}$
6:     end for
7:     return $E'$
8: end function
```

Figure 4.7 shows an example of the MOShake procedure for a set E of two solutions with ten jobs and $q = 2$. Two random permutation operations are demonstrated for each solution. The first one is an exchange of batches in positions 2 and 4, and an insert of the batch in position 4 to position 5. In the second, two insert operations: the first one moves the batch in position 2 to position 5 and the batch in position 1 to position 3. All the permutation movements proposed in these neighborhoods generate feasible solutions for the HT scheduling problem.

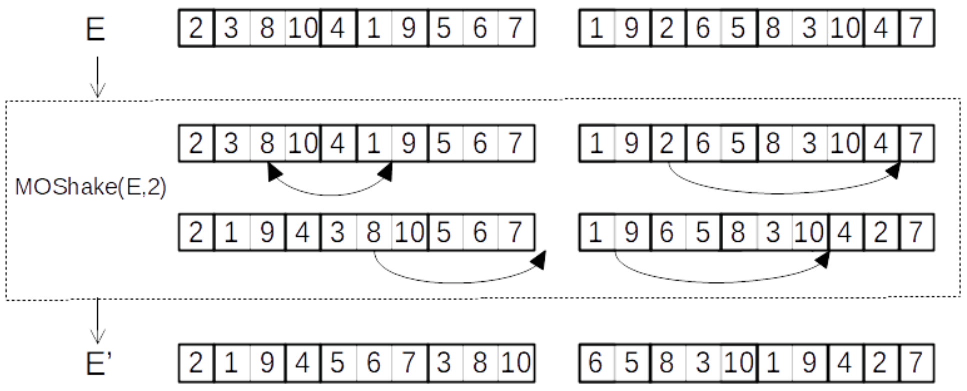


Figure 4.7 – Example of the MOShake procedure with parameters E and 2.

The set E has two solutions, represented as the permutation of six jobs. Two permutation operations ($q = 2$), chosen at random, are performed in each solution. For the first one, there is an exchange of batches in positions 2 and 4 and then an insert of the batch in position 4 to position 5. In the second solution, two insert operations are performed. The first one is moving the batch from position 2 to position 5. In the second one, the batch in position 1 is moved to position 3. This procedure defines the two new solutions of set E'.

Algorithm 4.5 elucidates how the multi-objective neighborhood change procedure works. As in the single objective version, a neighborhood q moves to $q + 1$ when there are no further improvements in the current search. Improvements are checked in function MOImprovement through the comparison of the current set of efficient points E and the new one, E', found by the exploration and exploitation procedures. An improvement is assumed true if at least one solution x' of E' is not dominated by any of the current efficient points E. The improvement leads to the update of the approximated Pareto front and restarts the search from the first neighborhood $q = 1$ (lines 3 and 4). The dominance comparison between the sets of solutions can be computationally demanding when they are too large (DUARTE et al., 2015). In those cases, the neighborhood change algorithm should have an efficient procedure to reduce the set of Pareto points without degrading the front (this operation is standard in many population-based metaheuristics such as NSGA-II (DEB et al., 2002) and SPEA2 (ZITZLER; LAUMANNS; THIELE, 2001). Nevertheless, this approach was not necessary in the HT scheduling resolution.

Algorithm 4.5 Pseudocode of multi-objective neighborhood change.

```
1: function MONeighborhoodChange($E, E', q$)
2:     if MOImprovement($E, E'$) then
3:         $q \leftarrow 1$
4:         $E \leftarrow$ Update($E, E'$)
5:     else
6:         $q \leftarrow q + 1$
7:     end if
8:     return $E, q$
9: end function
```

The descent search adopted in this work is based on first-improvement. Although most of the multi-objective VND algorithms found in literature use best-improvement (GEIGER, 2004), our preliminary tests indicated that the first-improvement strategy could lead to superior results in the considered problem, taking into account the execution time limits assumed (support information file available at Gomes (2020)).

Algorithm 4.6 shows the multi-objective VND based on first-improvement (MOVND-FI). Its parameters are the set of points E and the neighborhood size l_{max}. A set of exploited solutions is also maintained in variable D to avoid repeated search effort and to stop the method once all the points on E have been visited (respectively, lines 5 and 13). While there are still unvisited solutions in E, a random $x \in E$ is chosen (line 4), and then MOFirstImprovement inspects the neighborhood $N_l(x)$ until it finds a non-dominated solution, to add to the approximated Pareto set E. If no non-dominated points are found, function MONeighborhoodChange moves the neighborhood. In case of improvement, the new point x' replaces the current solution in a new local search (line 10).

The approximated Pareto front solutions found by the MOGVNS algorithm are then selected and polished.

Algorithm 4.6 Pseudocode of multi-objective first improvement VND.

```
1:  function MOVND-FI($E, l_{max}$)
2:      $D \leftarrow \emptyset$
3:      repeat
4:          $x \leftarrow$ SelectRandom($E \setminus D$)
5:          $D \leftarrow D \cup \{x\}$
6:          repeat
7:              $x' \leftarrow$ MOFirstImprovement($x, E, l$)
8:              $E, l \leftarrow$ MONeighborhoodChange($E, \{x'\}, l$)
9:              if l = 1 then
10:                 $x \leftarrow x'$
11:             end if
12:         until $l > l_{max}$
13:     until $E \setminus D = \emptyset$
14:     return $E$
15: end function
```

4.6 Solutions Selection

Given the computational effort required for the solution polishing, before the final step of the proposed matheuristic, N solutions are selected from the MOGVNS approximated Pareto front. The selection is based on a measure commonly used in multi-objective algorithms, the crowding distance (CD). This metric is used to compare the solutions' fitness in the non-dominated front, considering a criterion of diversity maintenance in the objective space. The CD value of a solution provides a density estimation of solutions surrounding it and is calculated considering the average distance of its neighboring solutions (DEB et al., 2002).

In Figure 4.8, the crowding distance value of the solution x (blue circle) is the average side length of the cuboid formed by its nearest neighbors, represented by the black dashed rectangle.

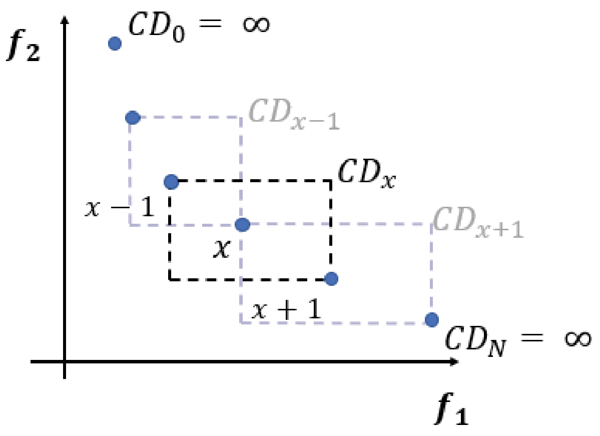


Figure 4.8 – Crowding distance calculation for a bidimensional objective space.

The crowding distance computation requires sorting the population according to each objective function value in ascending order of magnitude. For each objective function, the boundary solutions (solutions with the smallest function values) are assigned an infinite distance value. All other intermediate solutions are assigned a distance value equal to the absolute normalized difference in the function values of two adjacent solutions. This calculation is continued with other objective functions, and the overall crowding distance value is calculated as the sum of individual distance values corresponding to each objective (DEB et al., 2002).

Once the crowding distance of all the solutions of the MOGVNS approximated Pareto front is calculated, they are sorted in descending order and the N solutions with higher CD value (top of the sorted list) are selected for polishing.

4.7 Summary

To summarize the complete matheuristic execution, Figure 4.9 demonstrates how the approximated Pareto solutions evolve over the detailed steps. The first chart shows the two initial solutions generated through mathematical programming. One gives the optimal value for the total energy costs. In the second chart, these solutions appear as hashed circles as they have been improved and replaced by the solutions achieved from the MILP heuristics polishing. The red arrows indicate that the solutions have first moved towards the minimization of the total tardiness and then for the minimization of the total energy cost, in this case, only one of them. The third chart exemplifies the MOGVNS execution and its exploration of the search space mapping an approximated Pareto front. In the fourth step, five solutions of the front are selected, based on the crowding distance metric, for subsequent polishing. The fifth chart shows that one solution has improved its total energy costs from the solution polishing. This new solution dominates two solutions of the previous front that appear as hashed circles. The final approximated Pareto front has four solutions, as shown in the sixth chart, highlighted in a red box, and the matheuristic ends.

In the next chapter, the multi-objective matheuristic proposed is tested in real instances of the heat treatment scheduling case.

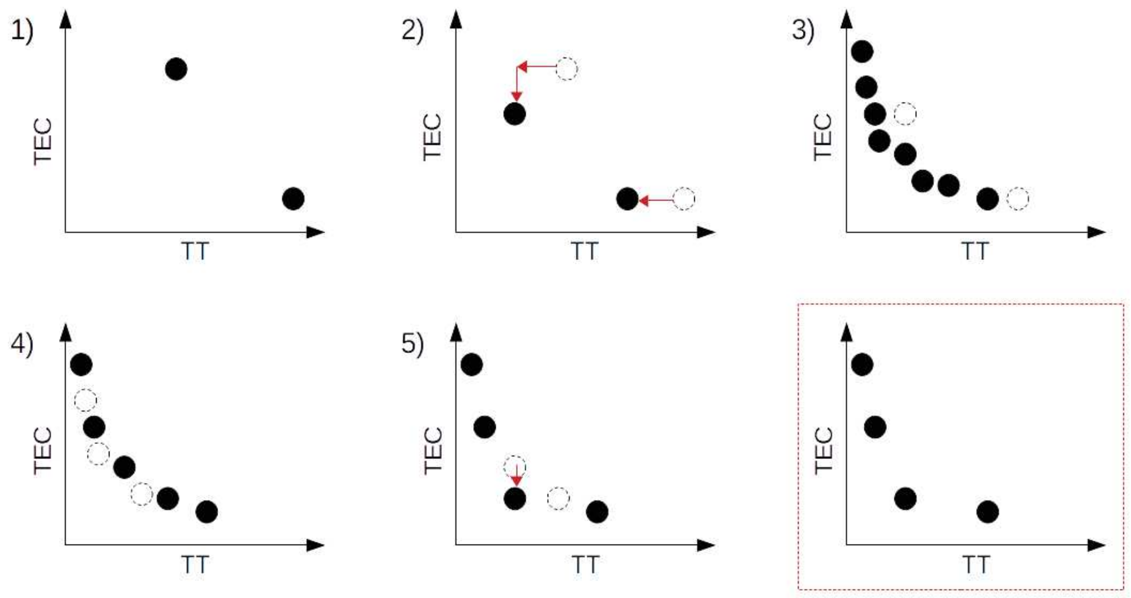


Figure 4.9 – Example of the approximated Pareto front evolution over the matheuristic steps.

5 Results and Discussion

5.1 Computational Experiments

The efficiency of the proposed multi-objective matheuristic, in the resolution of the heat treatment scheduling problem, was tested on 24 instances based on real data from the steel industry line. Each case represents six to seven days of production from the second semester of 2017. Figure 5.1 shows the number of jobs of the 24 instances. The smallest case has 37 jobs, and the biggest one has 112 jobs.

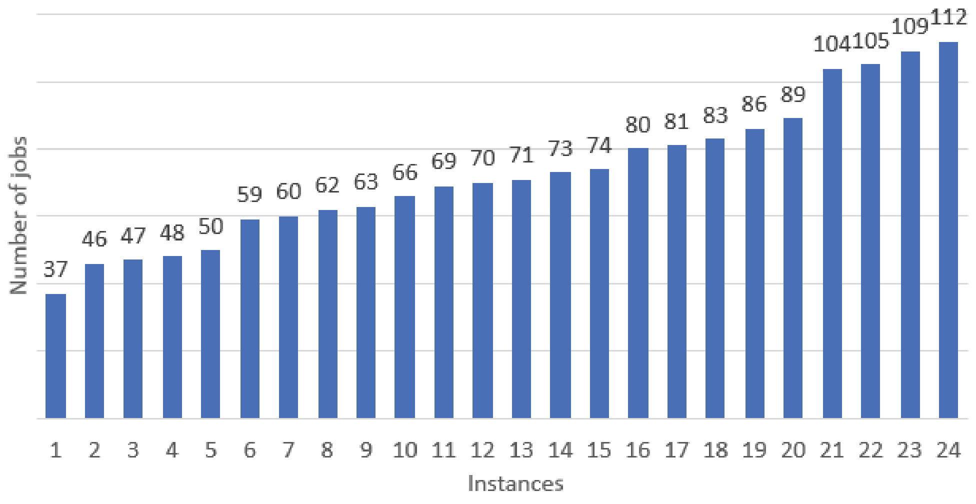


Figure 5.1 – Test instances with real data from second semester of 2017.

For each test instance, it is given the number of steel pieces to be produced, the material cycle time, the furnaces' treatment temperatures, and the due date (exemplified in Table 2.1). The parameters described in Section 2.2.2 were calculated from these inputs. The real sequence executed in the HT line is also known. It was used to validate the gains of the suggested methodology.

The proposed algorithms were executed in an 8GB RAM Ubuntu 20.04 LTS server and developed in C#, with .NET Core framework (MICROSOFT, 2019), following the software development standards of the steel industry. The solver used for MILP resolution was Gurobi 9.0.3 (GUROBI OPTIMIZATION, LLC, 2019) and the statistical analysis of the results was performed in R 3.6.3 (R CORE TEAM, 2016).

The experiments were split into four segments. First, the models presented in Section 4.2 were compared to define which one would be used to generate the matheuristic initial solutions. Second, the initial points were generated - by the selected models - and polished according to the methodology described in Section 4.4. The third part of the experiment focuses on the MOGVNS performance in terms of the configuration implemented and the initial solutions generated through mathematical programming. Finally, the last results of these experiments explore the solution polishing of the approximated Pareto fronts.

Five metrics are used to analyse the experiments and compare the algorithms:

- RT - runtime for the MILP solver (RT_{solver}), solution polishing (RT_{polish}), and MOGVNS (RT_{mogvns});
- GAP - final percentage gap between the lower and upper bounds of the MILP solver;
- PI - percentage improvement between two solutions according to each objective (PI_{TEC} and PI_{TT});
- HV - normalized hypervolume of the approximated Pareto front solutions;
- NS - number of non-dominated solutions at the approximated Pareto front.

Hypervolume is a metric used in multi-objective optimization to access the performance of optimizers regarding convergence and diversity. It was proposed by Zitzler and Thiele (1999) to stipulate the total volume existing between the approximated Pareto solutions and a reference point. A higher hypervolume indicates a better Pareto-front approximation. This metric was chosen because it does not depend on prior knowledge of the optimal Pareto front. The hypervolume values to be presented in this chapter are all normalized. The non-dominated points generated in the different executions were normalized between 0 and 1, and the estimated Nadir point was set as 1.1 for both objectives (the exact Nadir value is unknown).

The instances cost functions are not displayed to protect company information. Therefore, a percentage improvement (PI) measure was determined. This metric can be calculated for any solution S, taking as reference a solution S_0. The percentage improvement is calculated for both objectives by equations 5.1 and 5.2. The TT_0 and TEC_0 are, respectively, the total tardiness and total energy costs of the solution S_0.

$$PI_{TT}(\%) = \left(1 - \frac{TT}{TT_0}\right) 100. \tag{5.1}$$

$$PI_{TEC}(\%) = \left(1 - \frac{TEC}{TEC_0}\right) 100. \tag{5.2}$$

The following four sections detail these experiments' results. The expected gains for this methodology applied in the steel industry are presented at the end of the chapter.

5.2 Comparison of Mathematical Models

Two tests were performed to define which model - positional, or precedence - would be more appropriate to generate the two initial solutions of the HT scheduling problem.

As explained in Section 4.3, this step is based on a grouping strategy of the jobs, and they also use the single-objective version of the models. They are constructed by removing the objective function, constraints, and variables that are not related to the minimization goal. Therefore, the comparison presented next is based on the single-objective models that minimize the total energy costs or the total tardiness.

Each model ran five times for each objective resolution, and the MILP solver runtime was limited to two hours. For each model, it is compared the execution average runtime and the solver final gap.

Table 5.1 shows the comparison between the positional and precedence model in the minimization of the TEC objective. The first three columns, from left to right, have the instance identification name and size (n) and the number of jobs considered in the minimization of TEC (n_{TEC}) - after applying the grouping strategy. The two following sections show the average runtime, in seconds, and the percentage gap, respectively, for the positional and precedence models.

Table 5.1 – Comparison between the positional and precedence model in the minimization of the total energy cost objective.

Instance	n	n_{TEC}	Positional model		Precedence model	
			RT (s)	*GAP* (%)	*RT* (s)	*GAP* (%)
1	37	6	0.0542	0	**0.0069**	0
2	46	7	0.2042	0	**0.0061**	0
3	47	3	**0.0044**	0	0.0047	0
4	48	6	0.0702	0	**0.0031**	0
5	50	8	0.2097	0	**0.0069**	0
6	59	8	0.4331	0	**0.106**	0
7	60	5	0.0232	0	**0.0031**	0
8	62	7	0.1398	0	**0.0091**	0
9	63	6	0.046	0	**0.0044**	0
10	66	4	0.012	0	**0.0018**	0
11	69	6	0.1108	0	**0.0027**	0
12	70	10	1.3248	0	**0.0061**	0
13	71	14	71.3522	0	**0.0298**	0
14	73	4	0.0057	0	**0.0019**	0
15	74	14	237.0979	0	**0.0555**	0
16	80	8	0.2621	0	**0.007**	0
17	81	8	0.6846	0	**0.0085**	0
18	83	14	67.6489	0	**0.0225**	0
19	86	7	0.1403	0	**0.0033**	0
20	89	6	0.0553	0	**0.0079**	0
21	104	6	0.0918	0	**0.0058**	0
22	105	7	0.1982	0	**0.0174**	0
23	109	9	0.8962	0	**0.0307**	0
24	112	5	0.0298	0	**0.0018**	0

Table 5.1 attests that both models were able to solve the TEC minimization problem at its optimality, as they all reach a 0 gap within 2 hours of execution. It also can be observed that the grouping strategy can reduce the problem size significantly with instances that go from 3 to a maximum of 14 jobs. Except for instance 3, the precedence model showed a better runtime, as highlighted in bold. It was able to solve all problems in less than 1 second, while the positional model reached 237 seconds for the worst case, with an average runtime of 16 seconds. These results can be explained by the TEC single-objective model's reduced size, based on the TSP formulation.

The results were different for the total tardiness minimization, as shown by Table 5.2. The columns are the same from Table 5.1, except for the number of jobs considered in the minimization of TT (n_{TT}).

Table 5.2 – Comparison between the positional and precedence model in the minimization of the total tardiness objective.

Instance	n	n_{TT}	Positional model		Precedence model	
			RT (s)	*GAP* (%)	*RT* (s)	*GAP* (%)
1	37	8	0.0376	0	**0.0043**	0
2	46	13	1.1504	0	**0.338**	0
3	47	5	0.0256	0	**0.0018**	0
4	48	10	**65.9862**	0	201.8528	0
5	50	10	**39.6113**	0	132.6202	0
6	59	13	**893.8634**	0	7200	100
7	60	11	**14.8249**	0	63.8586	0
8	62	10	**31.2019**	0	120.5439	0
9	63	10	**46.5832**	0	82.3899	0
10	66	8	1.9029	0	**0.7467**	0
11	69	9	0.0517	0	**0.0047**	0
12	70	14	**4211.707**	0	7200	99.5
13	71	18	**7200**	100	**7200**	100
14	73	13	0.2159	0	**0.0379**	0
15	74	19	**7200**	**66.05**	**7200**	100
16	80	12	**341.9848**	0	7200	98.17
17	81	11	**64.5496**	0	515.2027	0
18	83	19	**7200**	92.27	**7200**	92.27
19	86	17	4.2663	0	**0.0633**	0
20	89	11	**266.9994**	0	2714.1802	0
21	104	11	**148.3003**	0	1585.8836	0
22	105	11	**81.9815**	0	941.6375	0
23	109	13	**1660.303**	0	7200	66.12
24	112	10	7.6249	0	**3.2823**	0

Comparing Table 5.1 and Table 5.2, it is possible to see that, for all test instances, the number of grouped jobs for the solution of the TT minimization is bigger than for TEC ($n_{TT} > n_{TEC}$). The fact that the number of grouped jobs for TT is larger results in longer runtimes. For 13 instances, the positional model runtime was smaller compared to the precedence model. For three instances, both models reached the solver execution time limit, and, in one of these cases, the positional model presented a better gap. Instances 6, 12, 16, and 23 were not solved by the precedence model, with relevant gap values. The results also indicate that the positional model performs better when the problem size increases, with a big difference in the average runtime compared to the precedence model, which can be related to its disjunctive constraints.

Given the results shown by Table 5.1 and 5.2, the single-objective precedence model was selected to generate the initial solution that minimizes the total energy cost objective. For the total tardiness objective, the positional model was selected due to its capacity to solve bigger problems, which gives the methodology better scalability.

5.3 Initial Solutions with Polishing

The second experiment investigated the matheuristic initial solutions generation through the selected mathematical models, and the polishing mechanism. The initial solution generated by the precedence model, for minimization of TEC, will be referred to as S_1. Solution S_2 is the one given by the positional model in the minimization of TT. They are represented by the hashed circles of Figure 5.2. The subsequent solution polishing generates the final initial solutions (black circles). The red arrows indicate the percentage improvements in S_1 and S_2 after the polishing.

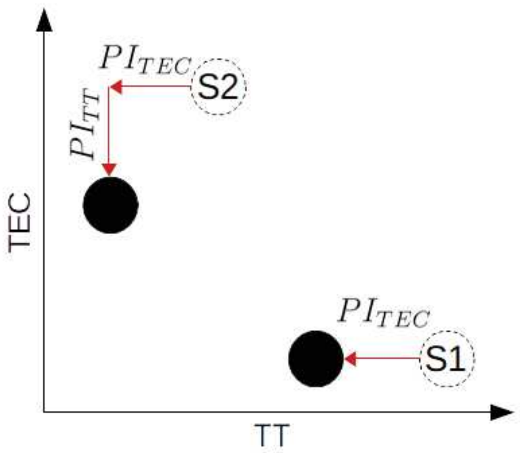


Figure 5.2 – Demonstration of the initial solutions generated by mathematical programming and subsequent polishing.

The MILP runs to generate S_1 and S_2 were limited to one hour, sufficient time to solve most of the problems, as presented in the previous section. Table 5.3 summarizes the outcomes of the 24 instances at this first step of the proposed matheuristic. The columns of the table are divided into three sections. The first one, from left to right, has the instance identification name and size. The second and third sections display the mathematical programming parameters and outputs for the generation of each solution. The first column of these sections shows the number of jobs considered in the single-objective minimization problems (n_{TEC} and n_{TT}). In the second and third columns it is shown the MILP solver runtime in seconds (RT_{solver}) and its final gap (GAP).

Table 5.3 – Summary of the MILP solver runs in the initial solutions generation for the 24 instances.

Instance	n	S_1			S_2		
		n_{TEC} (s)	RT_{solver} (s)	GAP (%)	n_{TT} (s)	RT_{solver} (s)	GAP (%)
1	37	6	0.006	0%	8	0.03	0%
2	46	7	0.0073	0%	13	1.68	0%
3	47	3	0.0082	0%	5	0.03	0%
4	48	6	0.0028	0%	10	92.24	0%
5	50	8	0.0032	0%	10	40.54	0%
6	59	8	0.1263	0%	13	1203.62	0%
7	60	5	0.0046	0%	11	21.93	0%
8	62	7	0.0158	0%	10	46.98	0%
9	63	6	0.0051	0%	10	69.75	0%
10	66	4	0.002	0%	8	2.76	0%
11	69	6	0.0031	0%	9	0.05	0%
12	70	10	0.0091	0%	14	3600.01	50%
13	71	14	0.0359	0%	18	3600.01	100%
14	73	4	0.0019	0%	13	0.24	0%
15	74	14	0.0905	0%	19	3600.01	68%
16	80	8	0.0066	0%	12	327.63	0%
17	81	8	0.0054	0%	11	69.25	0%
18	83	14	0.0202	0%	19	3600.01	92%
19	86	7	0.0026	0%	17	1.38	0%
20	89	6	0.0071	0%	11	250.74	0%
21	104	6	0.0057	0%	11	163.88	0%
22	105	7	0.0067	0%	11	84.85	0%
23	109	9	0.0221	0%	13	1817.11	0%
24	112	5	0.0022	0%	10	7.78	0%

Table 5.3 attests that the MILP solver can give the optimal solution S_1 for all instances within less than 1 second. In 4 cases (12, 13, 15, and 18), the solver reached the maximum execution time limit in the generation of S_2. Based on Table 5.2, instance 12 could have been solved with a zero-gap within 4211.7 seconds, however, this effort was left to the solution polishing mechanism.

The solution polishing maximum solver execution time, t_{max}, was limited to 1 minute to avoid a long iterative process. The maximum size of the fixed windows, h, was defined as 10 for the polishing of the total tardiness objective, and 20 for the TEC objective. In Table 5.2 it is possible to see that most of the instances with n_{TT} less than 10 were solvable within 60 seconds. For the TEC objective, this size was doubled, since the precedence model was able to solve through optimality instances up to 14 jobs.

To check the efficiency of the solution polishing, the initial solutions of the matheuristic were generated five times for each instance. Table 5.4 shows the polishing results. The columns of the table are divided into four sections. The first one is the instance identification name. The second section displays the average polishing runtime, in seconds, for both objectives. The polishing runs first to improve the total tardiness of S_1 and S_2 and then the total energy costs. The third and fourth sections show the percentage improvement in each objective reached for both solutions, S_1 and S_2, respectively.

Table 5.4 – Initial solutions polishing average results for the 24 instances.

Instance	RT_{polish} (s)		S_1		S_2	
	TT	TEC	PI_{TT}	PI_{TEC}	PI_{TT}	PI_{TEC}
1	26.28	0.31	11.3%	0%	0%	4.3%
2	211.4	60.05	1.9%	0%	0%	4.4%
3	301.69	0.05	2%	0%	3.4%	0.6%
4	269.64	20.5	13.9%	0%	20.1%	2%
5	78.18	5.01	1.6%	0%	25.5%	4.6%
6	169.88	60.09	3.6%	0%	25.7%	0.6%
7	205.64	3.27	19.2%	0%	19.3%	1.7%
8	898.71	41.75	1.5%	0%	2.1%	0.1%
9	251.73	1.28	2.7%	0%	5.2%	0.5%
10	1046.85	0.34	8.7%	0%	0.9%	0%
11	608.07	0.3	0%	0%	0%	5.4%
12	276.16	60.17	4.8%	0%	5.7%	1.9%
13	351.51	60.09	6.1%	0%	6.3%	1.5%
14	212.98	34.57	1.9%	0%	0%	8.1%
15	1553.53	60.17	0.2%	0%	2.1%	0.5%
16	1394.51	60.16	1.3%	0%	10.1%	0.2%
17	1054.88	2.81	0.8%	0%	7.3%	1.1%
18	786.72	60.21	2.1%	0%	11.1%	2.6%
19	484.01	60.23	1.5%	0%	0%	5.6%
20	1156.75	60.18	1.5%	0%	4.6%	0.9%
21	1374.64	5.11	1.2%	0%	12.9%	0.9%
22	958.14	0.65	3.6%	0%	6.8%	1.1%
23	3464.36	60.24	0.5%	0%	6.6%	0.9%
24	3762.57	2.79	7.5%	0%	8.7%	1.9%

The average runtime of the TT polishing, for both solutions, is 870 seconds, and for the TEC polishing is 30 seconds. The runtime for TEC has reached more than 1 minute for ten instances, indicating that the maximum solver execution time was reached - the process for S_1 is quick, given that the solution is already optimal (0% of PI_{TEC} for all instances). Therefore, an increase on t_{max} for future TEC polishing executions could be evaluated. Despite this, the polishing mechanism reduced the solutions' cost functions for most of the cases. The mean percentage improvements in S_1 for the total tardiness objective was 4.1%. The mean percentage improvements in S_2 for the TT was 9.7% and for TEC it was 2.1%. These results support the use of polishing to generate better initial solutions.

5.4 MOGVNS Performance

After the initial solutions were generated, 30 executions of the MOGVNS algorithm were performed. The multi-objective metaheuristic parameters were set based on preliminary experiments, which are described in a support information file (GOMES, 2020). Their values are as follows:

- Size of descent search neighborhood (l_{max}): it is the number of neighborhoods visited during the descent search, which are the four represented in Figure 4.6;
- Extent of shaking neighborhood (q_{max}): up to three random exchange or insert operations over the batches;
- Maximum execution time (t_{max}): the execution time of MOGVNS was modeled as a linear function of the instance size. The chosen value was $15 \times n$, which is time enough to run, at least, five descent searches of the MOVND.

For each execution of the MOGVNS, it was calculated the final approximated Pareto front normalized hypervolume. Table 5.5 shows the best results - in terms of hypervolume for each instance. The first two columns are the instances identification name and size, and the other columns show the MOGVNS runtime in minutes, and the number of non-dominated solutions in the final approximated Pareto front.

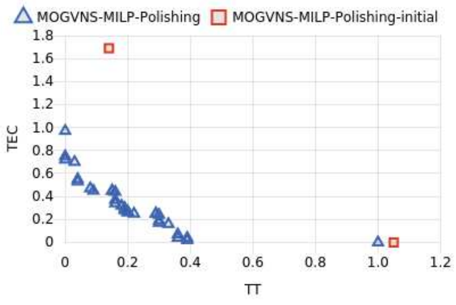
Results from this step of the matheuristic show that the MH runtime is lower than 30 minutes in the worst case. Therefore, even if the generation of the initial solution takes over one-hour each, the matheuristic total runtime is still an adequate time to perform a one week schedule. The number of efficient points found also varies from 1 to 30. A regression analysis shows that the number of jobs grouped by process characteristics - n_{TEC} - linearly correlates NS (R^2 of 70%), while the total number of jobs - n - does not influence it (R^2 of 0.7%). So, the number of solutions found in the approximated Pareto fronts are more strongly related to the product mixes. Consequently, a set of jobs with different process settings can be more difficult to solve, giving higher trade-offs between the objectives.

Figure 5.3 shows examples of two instances with the approximated Pareto front obtained by the algorithm: instances 15 and 18. In both charts, the blue triangles are the solutions found by the multi-objective metaheuristic, and the red squares are the two initial solutions generated by mathematical programming. This visualization demonstrates the importance of MOGVNS in the search space exploration. As the initial solutions that minimize TEC are optimal, there are no blue triangles with a lower TEC value. However, this is not the case for the TT, so the local search can further improve this objective. It is

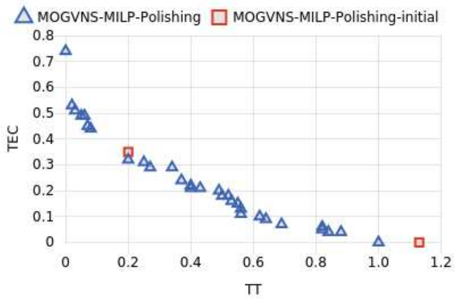
Table 5.5 – Summary of MOGVNS results for the 24 instances.

Instance	n	RT_{mogvns} (min)	NS
1	37	9.25	3
2	46	11.5	5
3	47	11.75	3
4	48	12	25
5	50	12.5	30
6	59	14.75	19
7	60	15	2
8	62	15.5	24
9	63	15.75	4
10	66	16.5	2
11	69	17.25	2
12	70	17.5	30
13	71	17.75	30
14	73	18.25	3
15	74	18.5	30
16	80	20	16
17	81	20.25	27
18	83	20.75	30
19	86	21.5	8
20	89	22.25	25
21	104	26	5
22	105	26.25	17
23	109	27.25	29
24	112	28	6

also possible to see the trade-off between objective functions for each solution.



(a) Instance 15.



(b) Instance 18.

Figure 5.3 – Examples of approximated Pareto fronts obtained by the MOGVNS for two instances.

Two comparative experiments were also conducted with the MOGVNS. The first, to evaluate the shaking and neighborhood configurations based on batches. The second, to estimate the benefit of using mathematical programming to generate its initial solutions.

In the first comparison, MOGVNS was executed 30 times with two different configurations. The first configuration, traditional in scheduling problems, uses two neighborhood sets, insert and exchange, and a shaking method that randomly does these operations over the jobs. Gomes, Ravetti, and Carrano (2021) implement this strategy for the solution of the HT scheduling problem. This configuration is referred as $N2$. The second configuration, proposed in this work, uses insert and exchange moves over the jobs and the sequence batches. The latter is also used in the shaking operations. This configuration is referred to as $N4$.

The hypervolume of the approximated Pareto fronts found by the two MOGVNS configurations were calculated for each execution. Figure 5.4 shows their distribution per instance. It is noticeable from the boxplots that configuration $N4$ presented a higher median hypervolume for most cases. For six instances, the median hypervolume reached is the same, and in other four problems, the observed difference was small (difference in the third decimal value).

To precisely compare the MOGVNS configurations, a statistical one-sided paired t-test was performed. The null hypothesis assumes that the mean hypervolume differences of configurations $N4$ and $N2$ are zero, and the alternative hypothesis assumes that the mean differences (μ_{N4} - μ_{N2}) are greater than zero (CAMPELO, 2018). The results indicated the rejection of the null hypothesis with a level of confidence of 95% (p-value=0.00032). It means that configuration $N4$ outperforms $N2$ with a relevant statistical hypervolume difference.

The four neighborhoods and the batches in the shaking method also improved the MOGVNS convergence - taking the normalized hypervolume calculated over the total execution time in seconds. Figure 5.5 shows how the solutions hypervolume changed over the algorithm iterations in two instances, 12 and 13. The blue curve shows the results from the $N4$ configuration (MOGVNS-N4) and the red from the $N2$ (MOGVNS-N2). In the first case (5.5a), configuration $N2$ seems to be trapped in local optimum solutions at the beginning of the iterations, which does not happen in $N4$. In the latter case (5.5b), configuration $N2$ took almost double of the time to reach the hypervolume of configuration $N4$. Despite that, both techniques present asymptotic convergence with quicker improvements in solutions over the first iterations and slower at the end.

To show the advantage of using mathematical programming to find good initial solutions, we implemented a different method. In this method, two heuristics define the initial solutions: the first is the well-known Earliest Due Date (EDD), and the second is a simple ordering of the jobs from the highest treatment temperatures to the lowest, which requires less energy during setup. The MOGVNS algorithm was initialized by these heuristics and the designed MILP and was executed 30 times for each test instance.

The boxplots of Figure 5.6 show the distribution per instance of the normalized hypervolumes calculated for both methods. The blue series represents the multi-objective

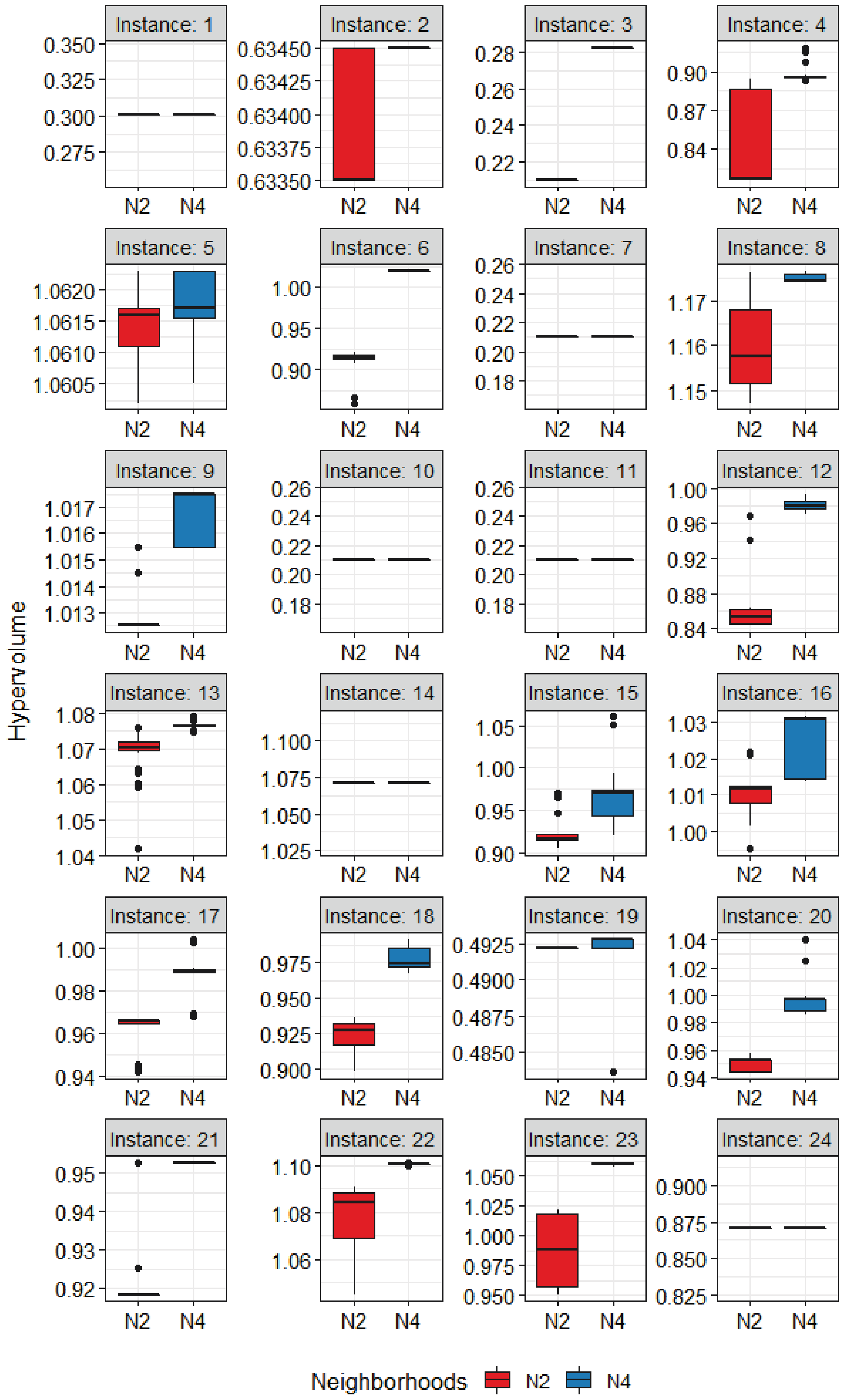
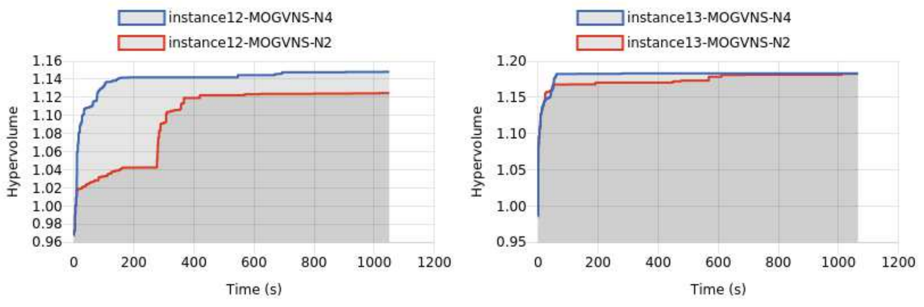


Figure 5.4 – Distribution per instance of the hypervolume calculated over 30 executions of the MOGVNS with two different configurations - ($N2$) and ($N4$).



(a) Instance 12. (b) Instance 16.

Figure 5.5 – Comparison of normalized hypervolume calculated over the total execution time (seconds) of MOGVNS algorithm with $N2$ and $N4$ configurations.

matheuristic, and the red series the comparative technique. The median hypervolume of the suggested methodology is greater than the metaheuristic alone for most of the instances. It indicates that the approximated Pareto solutions found by the multi-objective matheuristic present a better convergence and diversity. For instances 3 and 7, the hypervolume distribution is the same for both methods. For instances 6, 16, and 20, the median values of the proposed matheuristic are lower than the MOGVNS with simple heuristics.

A one-sided paired t-test was performed to make sure that the overall hypervolume differences are statistically significant. The null hypothesis assumes that the mean differences between the two sets are zero (CAMPELO, 2018). The results indicated the rejection of the null hypothesis with a level of confidence of 95% (p-value=0.01335). It means that the proposed matheuristic outperforms the simple MH with a relevant statistical difference in terms of the hypervolume performance metric.

Figure 5.7 presents two examples of approximated Pareto fronts found by the different initialization methods. They are from instances 6 and 23 and demonstrate how far the initial solutions from mathematical programming strategy (red squares) are from the solutions defined by the heuristics (green diamonds). As a result of the quality of the initial solutions, the matheuristic final approximated Pareto fronts (blue triangles) presented, in general, better diversity (Figure 5.7a) and convergence (Figure 5.7b). Figure 5.7b shows the MOGVNS-Heuristics solutions (orange circles) being dominated by the matheuristic front. These results justify the computational effort to find the initial points with the use of MP.

Figure 5.8 demonstrates the comparative convergence curves between the two techniques for instances 6 and 23. The blue curve shows the results from the proposed matheuristic (MOGVNS-FI-MILP) and the red from the MH with the simple heuristic initialization (MOGVNS-FI-Heuristics). They show that the matheuristic convergence is faster due to the use of MILP in the first phase of optimization that prioritizes the

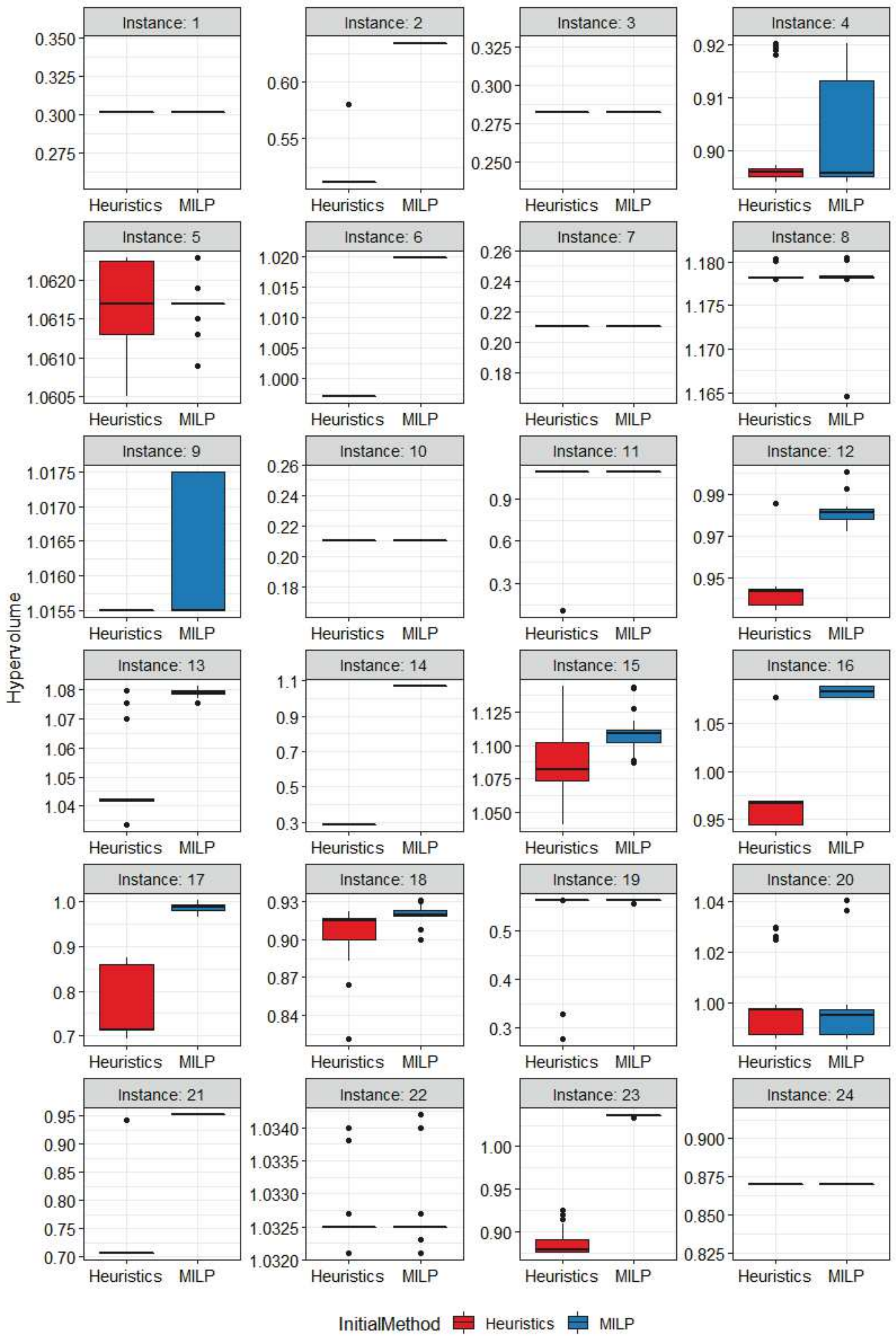
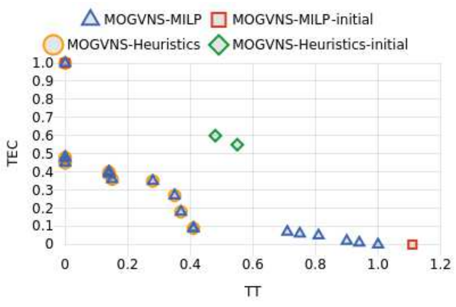
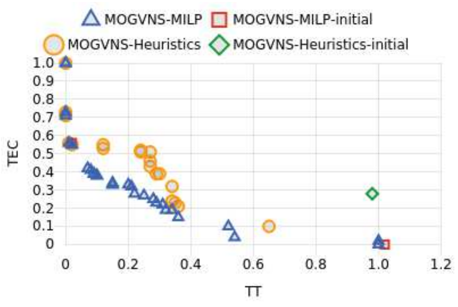


Figure 5.6 – Distribution per instance of the hypervolume calculated over 30 executions of the multi-objective matheuristic (MILP for initial solutions + MOGVNS) and MOGVNS with simple heuristics to generate the initial solutions.



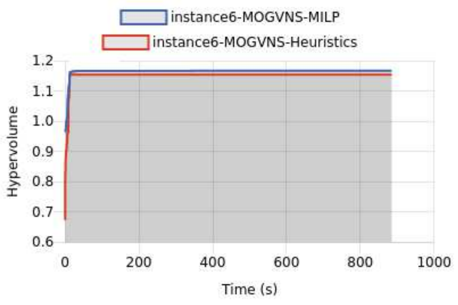
(a) Instance 6.



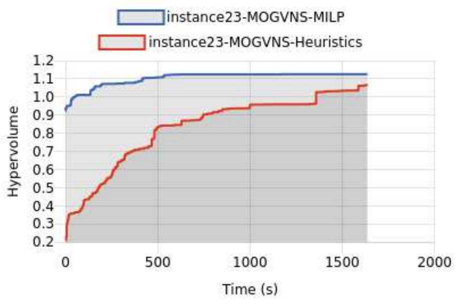
(b) Instance 23.

Figure 5.7 – Comparison of the approximated Pareto fronts found by the proposed matheuristic and by the simple heuristics to generate the initial solutions.

definition of good approximated Pareto extreme solutions. From the red curve, it is also possible to notice that the hypervolume reached by the only MOGVNS execution is smaller than the one from the combination of MP and MH. This result favors the conclusions shown above regarding the convergence of the methods.



(a) Instance 6.



(b) Instance 23.

Figure 5.8 – Comparison of normalized hypervolume calculated over the total execution time (seconds) of MOGVNS algorithm with MILP initialization and simple heuristics.

5.5 Approximated Pareto Polishing

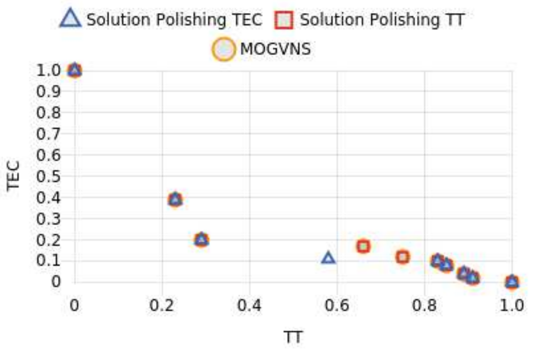
The final steps of the proposed multi-objective matheuristic are the MOGVNS crowding based selection, and the solution polishing application. The results from this section are from five complete matheuristic executions for each instance. Ten solutions were selected through the crowding distance to avoid the polishing of solutions with close function values. Table 5.6 shows, per instance, the final number of non-dominated solutions and the total runtime spent to try to improve them - first for the TT objective, and then for the TEC. The last column of the table shows if there was an improvement in the selected solutions hypervolume after the polishing.

Table 5.6 – Summary of the approximated Pareto fronts polishing for the 24 instances.

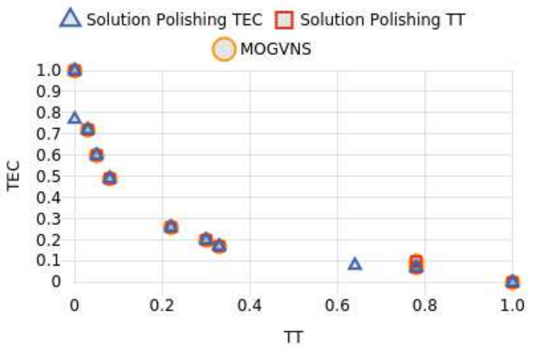
Instance	NS	RT_{polish} (min) TT	TEC	Improved HV?
1	3	0.39	0.01	N
2	5	7.36	0.12	N
3	3	42.7	0.71	N
4	10	45.34	0.76	N
5	10	36.22	0.6	N
6	10	11.35	0.19	N
7	2	6.67	0.11	N
8	10	105.03	1.75	N
9	4	16.52	0.28	N
10	2	18.89	0.31	N
11	2	8.74	0.15	N
12	10	42.89	0.71	Y
13	10	47.16	0.79	N
14	4	18.3	0.3	N
15	10	211.63	3.53	Y
16	10	113.49	1.89	Y
17	10	138.61	2.31	N
18	10	85.94	1.43	N
19	8	55.38	0.92	N
20	10	115.28	1.92	Y
21	5	120.33	2.01	N
22	10	441.34	7.36	N
23	10	358.86	5.98	Y
24	6	241.48	4.02	N

The mean runtime for the total tardiness and the total energy costs polishing was, respectively, 95.4 minutes, and 1.6 minutes. The execution time is proportional to the number of solutions to be polished. However, the number of batches in a sequence can also interfere with this result. Only five instances showed improvements in its hypervolume after the polishing. Figure 5.9 shows the approximated Pareto fronts for the five instances - 12, 15, 16, 20, and 23 - that presented improvement after the polishing. The orange circles represent the original solutions generated by the MOGVNS and selected through crowding distance. The red squares represent the solutions that went through the TT polishing, and the blue triangles are the solutions after going through the TEC polishing.

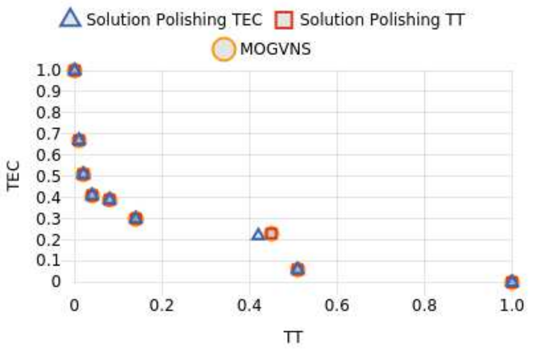
In all five cases, the TT polishing did not improve any solution. Notwithstanding, the TEC polishing improved the total energy cost of some solutions, which also upgraded their total tardiness. These results can be further explored in the future by executing the TEC polishing before the TT. It may support better outcomes. Otherwise, only the TEC polishing application is justified in the proposed method.



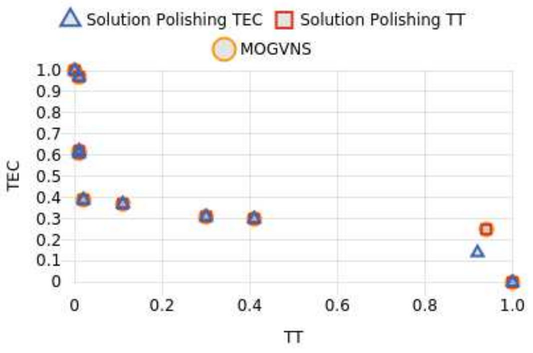
(a) Instance 12.



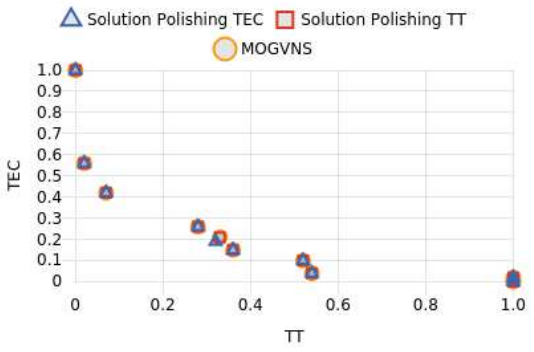
(b) Instance 15.



(c) Instance 16.



(d) Instance 20.



(e) Instance 23.

Figure 5.9 – Approximated Pareto front after polishing of the five instances that showed improvement.

5.6 Gains for the Steel Industry

This section focuses on estimating this work's benefits for the steel industry. The goal is to check if the alternative schedules found by the proposed multi-objective matheuristic are better than the ones executed in practice. Therefore, the approximated Pareto front extreme points were compared to the executed plans. The maximum improvement for the total tardiness objective (PI_{TT}^{max}) was calculated from the solution of minimum TT. The same value was calculated for the total energy costs (PI_{TEC}^{max}) from the solution of minimum TEC.

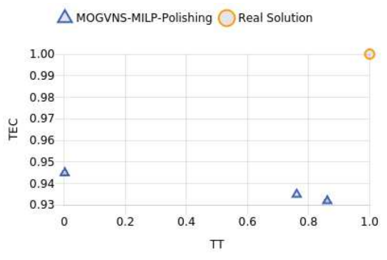
Table 5.7 shows the best results - in terms of hypervolume for each instance. The first column is the instance identification followed, respectively, by the maximum percentage improvement found for both objectives, (PI_{TEC}^{max} and PI_{TT}^{max}), in comparison to the solution executed in the HT line.

Table 5.7 – Expected gains for the application of the multi-objective matheuristic in the HT production line.

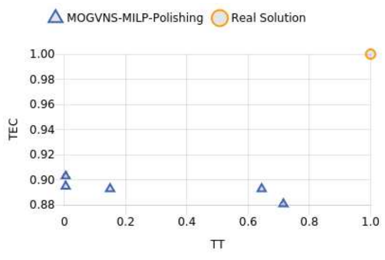
Instance	PI_{TEC}^{max}	PI_{TT}^{max}
1	6.8%	100%
2	11.9%	100%
3	1.8%	50.1%
4	6.2%	52.7%
5	4.5%	67.4%
6	2.9%	69.9%
7	4.2%	89.1%
8	6.8%	73.5%
9	2.3%	41.2%
10	2.2%	91.4%
11	13.3%	-
12	6.6%	56.3%
13	7.1%	99.5%
14	11.6%	100%
15	7.6%	85.2%
16	2.3%	71.8%
17	6.5%	77.9%
18	10.5%	73.1%
19	13.7%	100%
20	2.5%	56.1%
21	4.7%	74.1%
22	4.1%	80%
23	7.4%	83.5%
24	6.9%	92.9%

Regarding the maximum percentage improvement in objectives, observed gains are between 1.8% and 13.7% for the total energy costs and 41.2% up to 100% regarding the total tardiness (100% meaning TT of 0h after optimization). In instance 11, both real sequence and optimized sequence had no tardiness. Table 5.7 shows that the median gain for TT is 74.1% and the mean improvements in TEC are 6.4%. These values are relevant in practice and represent a substantial annual reduction for the heat treatment production line.

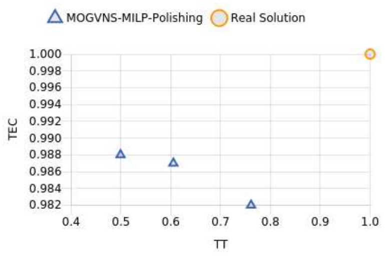
Figures 5.10 and 5.11 present all the solutions found by the proposed matheuristic in comparison to the company real executed sequence. The orange circles represent the real schedule solution, and the axes are on percentage. The approximated Pareto fronts found by the matheuristics are an improvement in the heat treatment schedule planning, as the real sequence executed is dominated for several of the solutions found in all instances.



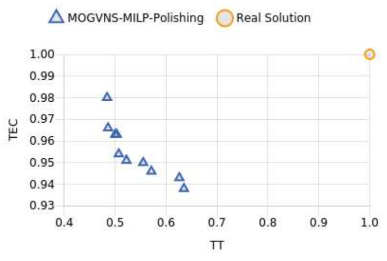
(a) Instance 1.



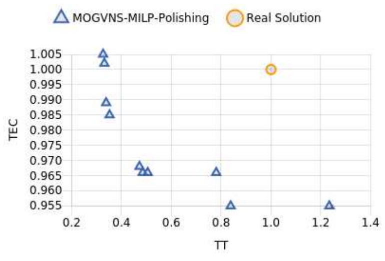
(b) Instance 2.



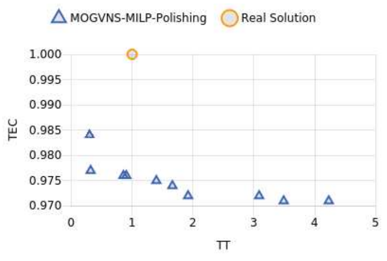
(c) Instance 3.



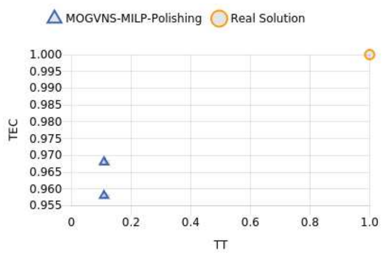
(d) Instance 4.



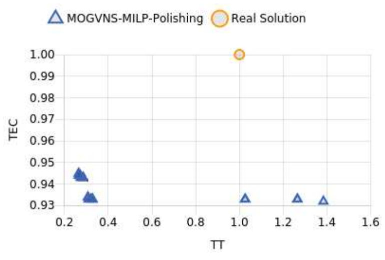
(e) Instance 5.



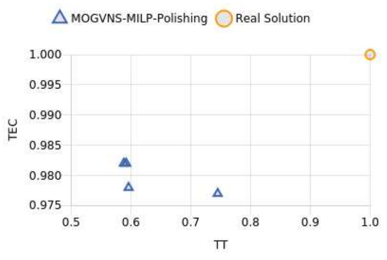
(f) Instance 6.



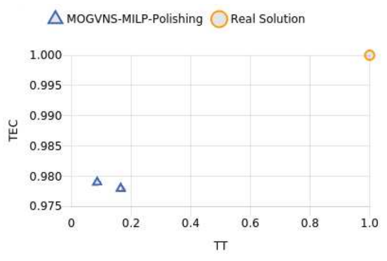
(g) Instance 7.



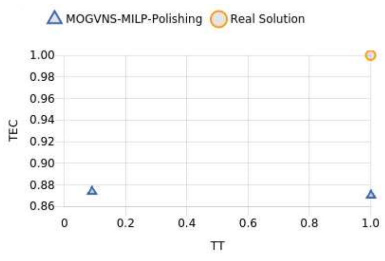
(h) Instance 8.



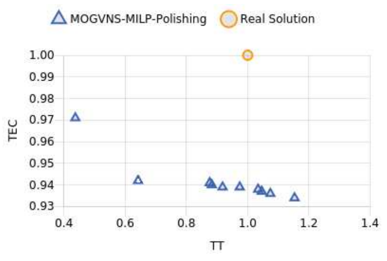
(i) Instance 9.



(j) Instance 10.

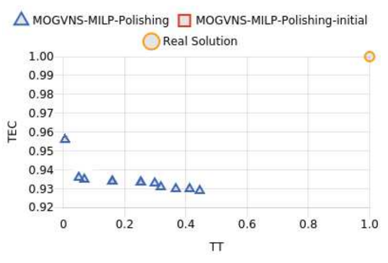


(k) Instance 11.

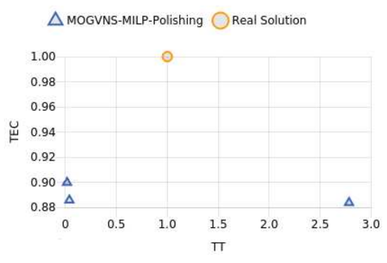


(l) Instance 12.

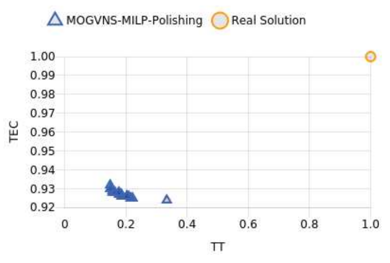
Figure 5.10 – Approximated Pareto fronts of instances 1 to 12 with the real company sequence execution.



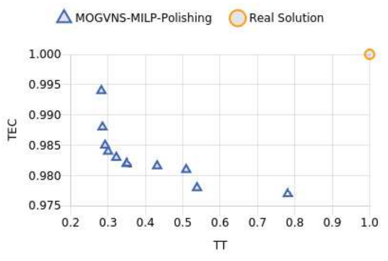
(a) Instance 13.



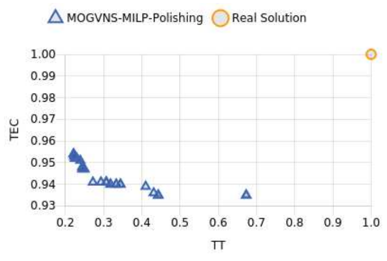
(b) Instance 14.



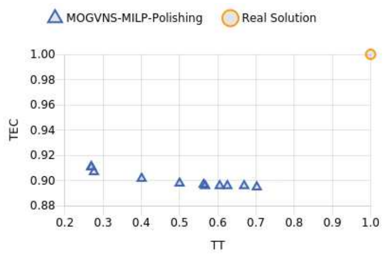
(c) Instance 15.



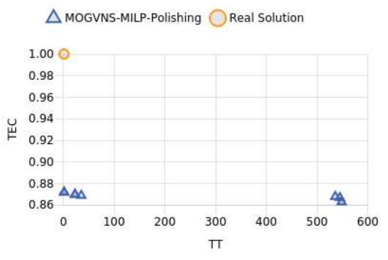
(d) Instance 16.



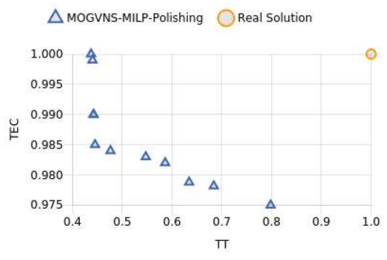
(e) Instance 17.



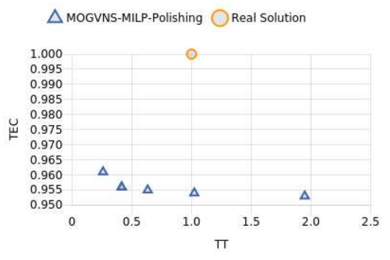
(f) Instance 18.



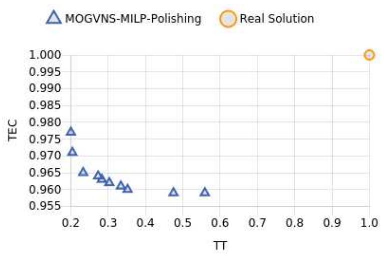
(g) Instance 19.



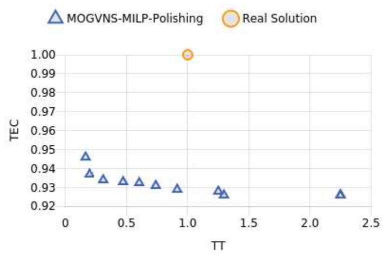
(h) Instance 20.



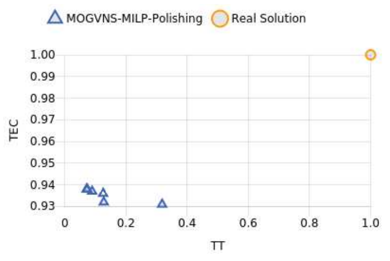
(i) Instance 21.



(j) Instance 22.



(k) Instance 23.



(l) Instance 24.

Figure 5.11 – Approximated Pareto fronts of instances 13 to 24 with the real company sequence execution.

All the experimental data and a support information file are publicly available at Gomes (2020).

6 Conclusions

6.1 Synthesis and main results

This research investigated a real scheduling production problem in a steel industry heat treatment (HT) line, which concerns the minimization of the total tardiness and total energy costs. It is a critical pilot study for a multinational company that has invested in developing an integrated scheduling system with optimization procedures to reduce its costs and increase its competitiveness worldwide.

A multi-objective matheuristic is proposed for the solution of the HT scheduling problem. The combination of metaheuristics and mathematical programming is a field of increased interest not explored enough for multi-objective problems. Therefore, this research examined the sequential execution of an exact solver and a multi-objective metaheuristic. The first generates two initial solutions to map the Pareto front extremes. The second scrutinizes the search space for non-dominated solutions that show compromises between the TT and TEC objectives.

The definition of the HT mathematical model was based on well-known formulations of no-wait flow shop scheduling problems. The intention was to find a model that could represent the heat treatment line characteristics and restrictions. For that, the classical parameters of NWFSP-SDST formulations were adapted to the heat treatment line reality, being extendable to similar problems. The steel company is also adopting this approach for the rolling mill and finishing lines.

For the success of the matheuristic, it was important to define a model that could be partially solved by an exact method during the generation of the initial solutions. For that, two different formulations were analyzed: a positional-based and a precedence-based model. They were tested in two HT single-objective problems: minimization of TT and TEC separately. The computational results revealed that the precedence model was capable of solving the TEC problem in less than 1 second for all instances, outperforming the positional model. Differently, the latest was more suitable for the TT problem resolution, presenting a better runtime and gap within the solver two hours execution limit. The positional model also presented better results for the TT minimization when the instances increased.

The HT scheduling instances are too big to be solved exactly. Therefore, to support the selected models' resolution, this work applied a strategy to reduce the instances' number of jobs, grouping them in batches. The jobs from the same batch are executed immediately after the other, without any setup. The experiments showed that this simplification proved to be essential for the generation of the initial solutions, the MOGVNS performance, and the polishing mechanism.

The solution polishing strategy proposed was based on a fix-and-optimize approach. It showed expressive improvements in the initial solutions generated by the MILP solver and a few in the final approximated Pareto fronts found by the local search metaheuristic. Therefore, this technique could be explored in different settings and at distinct stages of the matheuristic. The solution polishing has the potential to improve the scheduling optimization scalability. Thus, other practical scheduling problems could rely on the same approach.

Using mathematical programming to generate the initial solutions for the MOGVNS algorithm allows improvements in the final approximated Pareto fronts of the MH in terms of convergence and diversity. Different from population-based metaheuristics, local search algorithms highly benefit from good quality initial solutions. Therefore, the computational efforts of the multi-objective matheuristic first stages - with MILP solvers - are justified.

The computational simulations showed that the steel industry could gain, on average, almost 7% of reduction in the HT line total energy costs and 80% in its total tardiness by applying the proposed methodology. They also endorsed the compromise of the two objectives, sustaining the multi-objective approach.

6.2 Future work

Based on the synthesis and analysis of the results presented, some complementary topics could be considered for deeper investigation of the HT scheduling problem:

- Evaluation of different polishing configurations (e.g., fix-and-optimize window size, maximum iteration execution time) to further improve the initial solutions, the approximated Pareto fronts polishing and the final methodology scalability (considering instances that are not so reduced by the grouping strategy);
- Study of an integrated multi-objective matheuristic (e.g., using the fix-and-optimize approach as a MOGVNS neighborhood) to upgrade the final solutions hypervolume;
- Analyze the furnace thermodynamics giving an overall sequence temperature profile, and consider an objective or neighborhood rule to prevent aggressive temperature changes;
- Adaptation of the mathematical models and methodology to solve the problem with different scheduling characteristics (e.g., including the release date restriction and adding other objectives, such as minimization of the total completion time).

In the integrated steel industry broad planning context, other trending approaches could be considered in future works, such as the integration between the individual production lines scheduling and the corporate macro production planning, and the use of machine learning techniques (e.g., neural networks) to model the problem in a scenario of constant changes in the clients' demands that require more production flexibility and faster adaptation of the production planning and scheduling.

6.3 Publications

The following paper, produced during this thesis, was published in a scientific journal:

GOMES, Ana Cristina Lima; RAVETTI, Martín Gómez; CARRANO, Eduardo G. Multi-objective matheuristic for minimization of total tardiness and energy costs in a steel industry heat treatment line. en. **Computers & Industrial Engineering**, v. 151, Jan. 2021.

The following paper was published and presented in a national conference of Operational Research:

GOMES, Ana Cristina Lima; CARRANO, Eduardo Gontijo. A Multiobjective Approach for Scheduling a Quenching and Tempering Production Line. en. In: XLIX Simpósio Brasileiro de Pesquisa Operacional. [S.l.: s.n.], 2017. P. 1285–1295.

References

ALLAHVERDI, A. The third comprehensive survey on scheduling problems with setup times/costs. **European Journal of Operational Research**, v. 246, n. 2, p. 345–378, Oct. 2015.

ALLAHVERDI, Ali. A survey of scheduling problems with no-wait in process. en. **European Journal of Operational Research**, v. 255, n. 3, p. 665–686, Dec. 2016.

ALLAHVERDI, Ali; AYDILEK, Harun; AYDILEK, Asiye. No-wait flowshop scheduling problem with two criteria; total tardiness and makespan. en. **European Journal of Operational Research**, v. 269, n. 2, p. 590–601, Sept. 2018.

ANJANA, V.; SRIDHARAN, R.; RAM KUMAR, P. N. Metaheuristics for solving a multi-objective flow shop scheduling problem with sequence-dependent setup times. en. **Journal of Scheduling**, June 2019.

APPLEGATE, D. et al. On the Solution of Traveling Salesman Problems. en. **Documenta Mathematica**, p. 1–2, 1998.

ARABAMERI, S.; SALMASI, N. Minimization of weighted earliness and tardiness for no-wait sequence-dependent setup times flowshop scheduling problem. **Computers & Industrial Engineering**, v. 64, n. 4, p. 902–916, Apr. 2013.

BASSEUR, M. et al. Cooperation between Branch and Bound and Evolutionary Approaches to Solve a Bi-objective Flow Shop Problem. en. In: ________. **Experimental and Efficient Algorithms**. s.l.: Springer Berlin Heidelberg, 2004. (Lecture Notes in Computer Science), p. 72–86.

BIANCO, L.; DELL'OLMO, P.; GIORDANI, S. Flow Shop No-Wait Scheduling With Sequence Dependent Setup Times And Release Dates. **INFOR: Information Systems and Operational Research**, v. 37, n. 1, p. 3–19, Feb. 1999.

BLUM, C.; PUCHINGER, J., et al. Hybrid metaheuristics in combinatorial optimization: A survey. **Applied Soft Computing**, v. 11, n. 6, p. 4135–4151, Sept. 2011.

BLUM, C.; ROLI, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. **Acm Computing Surveys**, p. 268–308, 2003.

CAMPELO, Felipe. **Lecture Notes on Design and Analysis of Experiments**. [S.l.: s.n.], 2018. Accessed 26 February 2020. Available from: <https://github.com/fcampelo/Design-and-Analysis-of-Experiments>.

CAMPUZANO, Giovanni; OBREQUE, Carlos; AGUAYO, Maichel M. Accelerating the Miller–Tucker–Zemlin model for the asymmetric traveling salesman problem. en. **Expert Systems with Applications**, v. 148, p. 113–229, June 2020.

CHEN, Jing-fang; WANG, Ling; PENG, Zhi-ping. A collaborative optimization algorithm for energy-efficient multi-objective distributed no-idle flow-shop scheduling. en. **Swarm and Evolutionary Computation**, v. 50, p. 100557, Nov. 2019.

CHIEN, C.; LIAO, T. W.; DOU, R. Soft computing for smart production to empower industry 4.0. **Applied Soft Computing**, v. 68, p. 833–834, July 2018.

CROCE, F. D.; GROSSO, A.; SALASSA, F. A matheuristic approach for the two-machine total completion time flow shop problem. en. **Annals of Operations Research**, v. 213, n. 1, p. 67–78, Feb. 2014.

DANNA, E.; ROTHBERG, E.; PAPE, C. L. Exploring relaxation induced neighborhoods to improve MIP solutions. en. **Mathematical Programming**, v. 102, n. 1, p. 71–90, Jan. 2005.

DANTZIG, G.; FULKERSON, R.; JOHNSON, S. Solution of a Large-Scale Traveling-Salesman Problem. **Journal of the Operations Research Society of America**, v. 2, n. 4, p. 393–410, 1954. Publisher: INFORMS.

DEB, K. et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, v. 6, n. 2, p. 182–197, Apr. 2002.

DOSSETT, J. L.; BOYER, H. E. **Practical Heat Treating**. 2nd. Materials Park, Ohio: ASM International, Mar. 2006.

DUARTE, A. et al. Multi-objective variable neighborhood search: an application to combinatorial optimization problems. en. **Journal of Global Optimization**, v. 63, n. 3, p. 515–536, Nov. 2015.

DUMITRESCU, I.; STÜTZLE, T. Combinations of Local Search and Exact Algorithms. en. In: ________. **Applications of Evolutionary Computing**. s.l.: Springer Berlin Heidelberg, 2003. (Lecture Notes in Computer Science), p. 211–223.

EHRGOTT, M.; GANDIBLEUX, X. Hybrid Metaheuristics for Multi-objective Combinatorial Optimization. In: BLUM, C. et al. (Eds.). **Hybrid Metaheuristics: An Emerging Approach to Optimization**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. P. 221–259.

FELTL, H.; RAIDL, G. R. An Improved Hybrid Genetic Algorithm for the Generalized Assignment Problem. In: PROCEEDINGS of the 2004 ACM Symposium on Applied Computing. New York, NY, USA: ACM, 2004. (SAC ’04), p. 990–995.

FISCHETTI, M.; LODI, A. Local branching. en. **Mathematical Programming**, v. 98, n. 1-3, p. 23–47, Sept. 2003.

FLESZAR, K.; CHARALAMBOUS, C.; HINDI, K. S. A variable neighborhood descent heuristic for the problem of makespan minimisation on unrelated parallel machines with setup times. en. **Journal of Intelligent Manufacturing**, v. 23, n. 5, p. 1949–1958, Oct. 2012.

FONSECA, Gabriela B.; NOGUEIRA, Thiago H.; RAVETTI, Martín Gómez. A hybrid Lagrangian metaheuristic for the cross-docking flow shop scheduling problem. **European Journal of Operational Research**, v. 275, n. 1, p. 139–154, May 2019.

GANDIBLEUX, X.; FREVILLE, A. Tabu Search Based Procedure for Solving the 0-1 MultiObjective Knapsack Problem: The Two Objectives Case. en. **Journal of Heuristics**, v. 6, n. 3, p. 361–383, Aug. 2000.

GEIGER, M. J. Randomised Variable Neighbourhood Search for Multi Objective Optimisation. **4th EU/ME: Design and Evaluation of Advanced Hybrid Meta-heuristics;** 2004.

GOMES, Ana Cristina Lima. **GitHub repository for: Multi-objective matheuristic for minimization of total tardiness and energy costs in a steel industry heat treatment line**. [S.l.: s.n.], 2020. Available from: <https://github.com/anaclg/mo-matheuristic-nwfsp-sdst>.

GOMES, Ana Cristina Lima; CARRANO, Eduardo Gontijo. A Multiobjective Approach for Scheduling a Quenching and Tempering Production Line. en. In: XLIX Simpósio Brasileiro de Pesquisa Operacional. [S.l.: s.n.], 2017. P. 1285–1295.

GOMES, Ana Cristina Lima; RAVETTI, Martín Gómez; CARRANO, Eduardo G. Multi-objective matheuristic for minimization of total tardiness and energy costs in a steel industry heat treatment line. en. **Computers & Industrial Engineering**, v. 151, Jan. 2021.

GRAHAM, R. L. et al. Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey. In: HAMMER, P. L.; JOHNSON, E. L.; KORTE, B. H. (Eds.). **Annals of Discrete Mathematics**. s.l.: Elsevier, Jan. 1979. P. 287–326.

GUROBI OPTIMIZATION, LLC. **Gurobi Optimizer Reference Manual**. [S.l.: s.n.], 2019. Accessed 15 October 2019. Available from: <https://www.gurobi.com/wp-content/plugins/hd_documentations/documentation/8.1/refman.pdf>.

HADERA, H. et al. Optimization of steel production scheduling with complex time-sensitive electricity cost. **Computers & Chemical Engineering**, v. 76, p. 117–136, May 2015.

HANSEN, P. et al. Variable Neighborhood Search. In: GENDREAU, M.; POTVIN, J. (Eds.). **Handbook of Metaheuristics**. Boston, MA: Springer US, 2010. P. 61–86.

HAOUARI, M.; LADHARI, T. A branch-and-bound-based local search method for the flow shop problem. en. **Journal of the Operational Research Society**, v. 54, n. 10, p. 1076–1084, Oct. 2003.

HARJUNKOSKI, I. et al. Scope for industrial applications of production scheduling models and solution methods. **Computers & Chemical Engineering**, v. 62, p. 161–193, Mar. 2014.

JAHUIRA, C. A. R. Hybrid Genetic Algorithm with Exact Techniques Applied to TSP. In: SECOND International Workshop on Intelligent Systems Design and Application. Atlanta, GA, USA: Dynamic Publishers, Inc., 2002. P. 119–124.

JIANG, Shenglong et al. A bi-layer optimization approach for a hybrid flow shop scheduling problem involving controllable processing times in the steelmaking industry. en. **Computers & Industrial Engineering**, v. 87, p. 518–531, Sept. 2015.

JOURDAN, L.; BASSEUR, M.; TALBI, E. Hybridizing exact methods and metaheuristics: A taxonomy. **European Journal of Operational Research**, v. 199, n. 3, p. 620–629, Dec. 2009.

KOMAKI, M.; MALAKOOTI, B. General variable neighborhood search algorithm to minimize makespan of the distributed no-wait flow shop scheduling problem. en. **Production Engineering**, v. 11, n. 3, p. 315–329, June 2017.

KOSTIKAS, K.; FRAGAKIS, C. Genetic Programming Applied to Mixed Integer Programming. en. In: ________. **Genetic Programming**. s.l.: Springer Berlin Heidelberg, 2004. (Lecture Notes in Computer Science), p. 113–124.

LEI, D. Multi-objective production scheduling: a survey. en. **The International Journal of Advanced Manufacturing Technology**, v. 43, n. 9, p. 9–26, Oct. 2008.

LEI, Deming. Variable neighborhood search for two-agent flow shop scheduling problem. en. **Computers & Industrial Engineering**, v. 80, p. 125–131, Feb. 2015.

LI, W. **Production Scheduling in Integrated Steel Manufacturing**. 2014. Theses and Dissertations – University of Wisconsin-Milwaukee.

LIN, Shih-Wei; LU, Chung-Cheng; YING, Kuo-Ching. Minimizing the Sum of Makespan and Total Weighted Tardiness in a No-Wait Flowshop. **IEEE Access**, v. 6, p. 78666–78677, 2018.

LIN, Shih-Wei; YING, Kuo-Ching. Optimization of makespan for no-wait flowshop scheduling problems using efficient matheuristics. en. **Omega**, v. 64, p. 115–125, Oct. 2016.

LIU, G.; SONG, S.; WU, C. Some heuristics for no-wait flowshops with total tardiness criterion. **Computers & Operations Research**, v. 40, n. 2, p. 521–525, Feb. 2013.

LIU, Ming; YANG, Xuenan, et al. Energy-oriented bi-objective optimization for the tempered glass scheduling. en. **Omega**, v. 90, p. 101–995, Jan. 2020.

M'HALLAH, R. Minimizing total earliness and tardiness on a permutation flow shop using VNS and MIP. en. **Computers & Industrial Engineering**, v. 75, p. 142–156, Sept. 2014.

MANNE, A. S. On the Job-Shop Scheduling Problem. **Operations Research**, v. 8, n. 2, p. 219–223, 1960.

MICROSOFT. **Guia do .NET Core**. pt-br. [S.l.: s.n.], 2019. Accessed 22 February 2020. Available from: <https://docs.microsoft.com/pt-br/dotnet/core/>.

MILLER, C. E.; TUCKER, A. W.; ZEMLIN, R. A. Integer Programming Formulation of Traveling Salesman Problems. **Journal of the ACM**, v. 7, n. 4, p. 326–329, Oct. 1960.

MLADENOVIĆ, N.; HANSEN, P. Variable neighborhood search. **Computers & Operations Research**, v. 24, n. 11, p. 1097–1100, Nov. 1997.

MÖNCH, Lars; ROOB, Sebastian. A matheuristic framework for batch machine scheduling problems with incompatible job families and regular sum objective. en. **Applied Soft Computing**, v. 68, p. 835–846, July 2018.

MUCHA, Marcin; SVIRIDENKO, Maxim. No-Wait Flowshop Scheduling Is as Hard as Asymmetric Traveling Salesman Problem. en. In: ________. **Automata, Languages, and Programming**. Berlin, Heidelberg: Springer, 2013. (Lecture Notes in Computer Science), p. 769–779.

NADERI, B. et al. Multi-objective no-wait flowshop scheduling problems: models and algorithms. **International Journal of Production Research**, v. 50, n. 10, p. 2592–2608, May 2012.

NAGANO, M. S.; ARAÚJO, D. C. New heuristics for the no-wait flowshop with sequence-dependent setup times problem. en. **Journal of the Brazilian Society of Mechanical Sciences and Engineering**, v. 36, n. 1, p. 139–151, Jan. 2014.

NAGANO, M. S.; MIYATA, H. H. Review and classification of constructive heuristics mechanisms for no-wait flow shop problem. en. **The International Journal of Advanced Manufacturing Technology**, v. 86, n. 5, p. 2161–2174, Sept. 2016.

NOGUEIRA, Thiago Henrique et al. Analysis of mixed integer programming formulations for single machine scheduling problems with sequence dependent setup times and release dates. en. **Pesquisa Operacional**, v. 39, n. 1, p. 109–154, Jan. 2019.

ÖNCAN, Temel; ALTINEL, İ. Kuban; LAPORTE, Gilbert. A comparative analysis of several asymmetric traveling salesman problem formulations. en. **Computers & Operations Research**, v. 36, n. 3, p. 637–654, Mar. 2009.

PAN, C. A study of integer programming formulations for scheduling problems. **International Journal of Systems Science**, v. 28, n. 1, p. 33–41, Jan. 1997.

PIEHLER, J. Ein Beitrag zum Reihenfolgeproblem. de. **Unternehmensforschung**, v. 4, n. 3, p. 138–142, Sept. 1960.

PINEDO, M. L. **Scheduling: Theory, Algorithms, and Systems**. 4th. New York: Springer-Verlag, 2012.

PRZYBYLSKI, A.; GANDIBLEUX, X.; EHRGOTT, M. A two phase method for multi-objective integer programming and its application to the assignment problem with three objectives. **Discrete Optimization**, v. 7, n. 3, p. 149–165, Aug. 2010.

PUCHINGER, J.; RAIDL, G. R. Combining Metaheuristics and Exact Algorithms in Combinatorial Optimization: A Survey and Classification. en. In: ________. **Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach**. s.l.: Springer Berlin Heidelberg, 2005. P. 41–53.

R CORE TEAM. **R: A Language and Environment for Statistical Computing**. s.l.: s.n., 2016. Available from: <https://www.R-project.org/>.

RAIDL, G. R.; PUCHINGER, J. Combining (Integer) Linear Programming Techniques and Metaheuristics for Combinatorial Optimization. In: BLUM, C. et al. (Eds.). **Hybrid Metaheuristics: An Emerging Approach to Optimization**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. P. 31–62.

RONCONI, D.; BIRGIN, E. Mixed-Integer Programming Models for Flowshop Scheduling Problems Minimizing the Total Earliness and Tardiness. In: JUST-IN-TIME Systems. s.l.: Springer, Jan. 2012. P. 91–105.

SAMARGHANDI, H.; ELMEKKAWY, T. Y. Solving the no-wait flow-shop problem with sequence-dependent set-up times. **International Journal of Computer Integrated Manufacturing**, v. 27, n. 3, p. 213–228, Mar. 2014.

SCHALLER, Jeffrey. Scheduling a permutation flow shop with family setups to minimise total tardiness. **International Journal of Production Research**, v. 50, n. 8, p. 2204–2217, Apr. 2012. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207543.2011.575094.

SHAO, Weishi; PI, Dechang; SHAO, Zhongshi. A Pareto-Based Estimation of Distribution Algorithm for Solving Multiobjective Distributed No-Wait Flow-Shop Scheduling Problem With Sequence-Dependent Setup Time. **IEEE Transactions on Automation Science and Engineering**, v. 16, n. 3, p. 1344–1360, July 2019. Conference Name: IEEE Transactions on Automation Science and Engineering.

SHEN, Liji; GUPTA, Jatinder N. D.; BUSCHER, Udo. Flow shop batching and scheduling with sequence-dependent setup times. en. **Journal of Scheduling**, v. 17, n. 4, p. 353–370, Aug. 2014.

SILVA, J. D. L.; BURKE, E. K.; PETROVIC, S. An Introduction to Multiobjective Metaheuristics for Scheduling and Timetabling. In: METAHEURISTICS for Multiobjective Optimisation. s.l.: Springer, Berlin, Heidelberg, 2004. P. 91–129.

SIQUEIRA, E. C. de; SOUZA, M. J. F.; SOUZA, S. R. de. A Multi-objective Variable Neighborhood Search algorithm for solving the Hybrid Flow Shop Problem. en. **Electronic Notes in Discrete Mathematics**, v. 66, p. 87–94, Apr. 2018.

STAFFORD, E. F. On the Development of a Mixed-Integer Linear Programming Model for the Flowshop Sequencing Problem. en. **Journal of the Operational Research Society**, v. 39, n. 12, p. 1163–1174, Dec. 1988.

STAFFORD, E. F.; TSENG, F. T. Two models for a family of flowshop sequencing problems. **European Journal of Operational Research**, v. 142, n. 2, p. 282–293, Oct. 2002.

TA, Q. C.; BILLAUT, J.; BOUQUARD, J. Matheuristic algorithms for minimizing total tardiness in the m-machine flow-shop scheduling problem. en. **Journal of Intelligent Manufacturing**, v. 29, n. 3, p. 617–628, Mar. 2018.

TALBI, E. Combining metaheuristics with mathematical programming, constraint programming and machine learning. en. **Annals of Operations Research**, v. 240, n. 1, p. 171–215, May 2016.

TANG, L. et al. A review of planning and scheduling systems and methods for integrated steel production. **European Journal of Operational Research**, v. 133, n. 1, p. 1–20, Aug. 2001.

TEC-SCIENCE. **Quenching and tempering of steel**. en-US. [S.l.: s.n.], July 2018. Accessed 28 February 2021. Available from: <`https://www.tec-science.com/material-science/heat-treatment-steel/quenching-and-tempering/`>.

TIAN, Huixin; LI, Kun; LIU, Wei. A Pareto-Based Adaptive Variable Neighborhood Search for Biobjective Hybrid Flow Shop Scheduling Problem with Sequence-Dependent Setup Time. en. **Mathematical Problems in Engineering**, 2016.

TSENG, F. T.; STAFFORD, E. F.; GUPTA, J. N. D. An empirical analysis of integer programming formulations for the permutation flowshop. **Omega**, v. 32, n. 4, p. 285–293, Aug. 2004.

ULUNGU, E. L.; TEGHEM, J. The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems. **Foundations of Computing and Decision Sciences**, v. 20, p. 149–165, 1995.

WAGNER, H. M. An integer linear-programming model for machine scheduling. en. **Naval Research Logistics Quarterly**, v. 6, n. 2, p. 131–140, June 1959.

WANG, Fucai et al. Multi-Objective Parallel Variable Neighborhood Search for Energy Consumption Scheduling in Blocking Flow Shops. **IEEE Access**, v. 6, p. 68686–68700, 2018.

WANG, J. et al. Batch scheduling for minimal energy consumption and tardiness under uncertainties: A heat treatment application. **CIRP Annals**, v. 65, n. 1, p. 17–20, Jan. 2016.

WANG, Jingjing et al. A Cooperative Algorithm for Energy-efficient Scheduling of Distributed No-wait Flowshop. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI). [S.l.: s.n.], Nov. 2017. P. 1–8.

WILSON, J. M. Alternative Formulations of a Flow-shop Scheduling Problem. en. **Journal of the Operational Research Society**, v. 40, n. 4, p. 395–399, Apr. 1989.

WISMER, D. A. Solution of the Flowshop-Scheduling Problem with No Intermediate Queues. **Oper. Res.**, v. 20, n. 3, p. 689–697, June 1972.

WORLD STEEL ASSOCIATION AISBL. **Global crude steel output increases by 3.4% in 2019**. en. [S.l.: s.n.], Jan. 2020. Accessed 27 January 2020. Available from: <http://www.worldsteel.org/media-centre/press-releases/2020/Global-crude-steel-output-increases-by-3.4--in-2019.html>.

WU, Xueqi; CHE, Ada. Energy-efficient no-wait permutation flow shop scheduling by adaptive multi-objective variable neighborhood search. en. **Omega**, p. 102–117, Sept. 2019.

YEH, Tsung-Su; CHIANG, Tsung-Che. An Improved Multiobjective Evolutionary Algorithm for Solving the No-wait Flow Shop Scheduling Problem. In: 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM). [S.l.: s.n.], Dec. 2018. P. 142–147.

ZHANG, R.; CHIONG, R. Solving the energy-efficient job shop scheduling problem: a multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. **Journal of Cleaner Production**, v. 112, p. 3361–3375, Jan. 2016.

ZHAO, S.; GROSSMANN, I. E.; TANG, L. Integrated scheduling of rolling sector in steel production with consideration of energy consumption under time-of-use electricity prices. **Computers & Chemical Engineering**, v. 111, p. 55–65, Mar. 2018.

ZHENG, Xu et al. Energy-efficient scheduling for multi-objective two-stage flow shop using a hybrid ant colony optimisation algorithm. **International Journal of Production Research**, v. 0, n. 0, p. 1–18, July 2019.

ZITZLER, E.; LAUMANNS, M.; THIELE, L. **SPEA2: Improving the Strength Pareto Evolutionary Algorithm**. s.l., 2001.

ZITZLER, E.; THIELE, L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. **IEEE Transactions on Evolutionary Computation**, v. 3, n. 4, p. 257–271, Nov. 1999.