University of São Paulo
São Carlos School of Engineering
Department of Production Engineering
Postgraduation Program in Production Engineering

Gustavo Alencar Rolim

Heuristics and stochastic local search for just-in-time scheduling of parallel machines against common restrictive due windows

São Carlos - São Paulo (SP)
2020

Gustavo Alencar Rolim

Heuristics and stochastic local search for just-in-time scheduling of parallel machines against common restrictive due windows

Master's thesis submitted to the Postgraduation Program in Production Engineering from São Carlos School of Engineering in fulfilment of the requirements for the Master's degree in Production Engineering.

Concentration Area: Processes and Operations Management.

Supervisor: Prof. Dr. Marcelo Seido Nagano

Corrected Version

São Carlos - São Paulo (SP)
2020

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da EESC/USP com os dados inseridos pelo(a) autor(a).

A982h

Alencar Rolim, Gustavo
Heuristics and stochastic local search for just-in-time scheduling of parallel machines against common restrictive due windows / Gustavo Alencar Rolim; orientador Marcelo Seido Nagano. São Carlos, 2020.

Dissertação (Mestrado) - Programa de Pós-Graduação em Engenharia de Produção e Área de Concentração em Processos e Gestão de Operações -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2020.

1. Scheduling. 2. Earliness and tardiness. 3. Common due windows. 4. Parallel machines. 5. Heuristics. 6. Stochastic local search. I. Título.

Eduardo Graziosi Silva - CRB - 8/8907

FOLHA DE JULGAMENTO

Candidato: Engenheiro **GUSTAVO ALENCAR ROLIM.**

Título da dissertação: "Heurísticas e busca local estocástica para o sequenciamento *just-in-time* de máquinas paralelas sujeitas a janelas de tempo restritivas comuns".

Data da defesa: 03/07/2020

Comissão Julgadora: **Resultado:**

Prof. Dr. **Marcelo Seido Nagano** Aprovado
(Orientador)
(Escola de Engenharia de São Carlos – EESC/USP)

Prof. Dr. **Bruno de Athayde Prata** APROVADO
(Universidade Federal do Ceará/UFC)

Prof. Dr. **Roberto Fernandes Tavares Neto** Aprovado
(Universidade Federal de São Carlos/UFSCar)

Coordenador do Programa de Pós-Graduação em Engenharia de Produção:
Prof. Dr. **Marcelo Seido Nagano**

Presidente da Comissão de Pós-Graduação:
Prof. Titular **Murilo Araujo Romero**

To those who make our existence worthwhile.

Acknowledgments

Over the course of the last two years, there were short and long meetings with people that contributed in distinct ways to this research. I express my sincere gratitude to Anselmo Pitombeira and Bruno Prata from Federal University of Ceará for encouraging me to pursue the academic career. During graduate school, I have developed the habit of telling them about my goals, successes, and failures. To Marcelo Nagano, who devoted his time and attention for providing me with guidance and for helping me to build solid foundations. I have been extraordinarily lucky to have a supervisor who still leaves the door open after almost two and a half years of endless interruptions. I am excited about the future path you all have pushed me toward.

I am also thankful to all the members of the Laboratory of Applied Operational Research (LAOR) from University of São Paulo, specially Hugo Miyata, Caio Tomazella, Edson Gonçalves, Fernando Siqueira, Viviane Junqueira, and Raínne Florisbelo for the invaluable conversations. The more experienced members Caio and Hugo were particularly important for teaching me the coding skills necessary to come up with this study.

To the gifted colleagues from Centro Acadêmico Armando Salles de Oliveira (CAASO) and from its photography group who are engaged in the fight for equality, respect, and education. Our union and dedication reflect a noble cause which is responsible for bringing hope to many other students. I strongly believe that we will be able to construct a better future for our country.

To my childhood friends from Fortaleza: Dimitri Rodrigues, Icaro Machado, Joel Freire, Pedro Barbosa, Pedro Dias, and Marcio Militão. I will always take the joy, the good moments, and our unique sense of humour everywhere I go. I want to extend a special thanks to Ana Carolina Veras and her family for the weekend trips to São Paulo and the support given for more than one and a half years. Despite our split paths, I will carry very meaningful memories. I am delighted to mention the great friends I met in São Carlos: Djaci Augusto, Shuji Ozawa, Renato Chavez, Rachel Marascalchi, Angelica Camargo, Paula Cavalcante, and the other companions from São Carlos School of Engineering. Nonato Portela, Rodrigo Máximo, and Monica Tanaka are also beloved

friends that were consistently by my side during this tough journey.

I could not have imagined the possibility of being here without the support and example of my family. My parents and grandparents have made it easy for me to follow my interests wherever they lead and supported me at every step of the process. I am exceptionally blessed to have grown up in the intellectual and scientific environment that I have always been surrounded by.

Finally, I acknowledge the financial support given by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) grants 306075/2017-2 and 430137/2018-4 and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) finance code 001.

"If the doors of perception were cleansed,
everything would appear to man as it is,
infinite"
William Blake

Abstract

Rolim, Gustavo Alencar. **Heuristics and stochastic local search for just-in-time scheduling of parallel machines against common restrictive due windows**. 125 p. Master Thesis – São Carlos School of Engineering, University of São Paulo, 2020.

This study deals with earliness and tardiness scheduling around common due dates and windows. Related problems share a strong practical importance due to the endogenous and exogenous cost structures that occur within just-in-time production systems. Despite of their significance, there are no recent research that holistically address the two main variants of the problems in terms of the common due date (window) approach. We fill this gap by surveying over 200 papers, which cover more than 40 years of scheduling literature and classifying algorithms for problems where the due date (window) is a given constraint as well as the ones where it is a decision variable. New notations and a comprehensive framework that includes a wide range of earliness and tardiness scheduling problems are introduced. Moreover, a total of 26 structural properties and the computational complexity of available algorithms for single, parallel, and flow-shop machine environments are summarized. From the literature review, we selected the just-in-time parallel machine scheduling problem against common restrictive due windows that is known to be NP-hard. Since there is no procedure available for solving instances with more than 40 jobs, we propose a family of heuristics and report promising results for instances with up to 500 jobs in size. Moreover, the best performing heuristic is combined with a stochastic local search (iterated greedy algorithm) to improve the solutions previously found.

Keywords: Scheduling; Earliness and Tardiness; Common due window; Parallel machines; Heuristics; Stochastic local search.

Resumo

Rolim, Gustavo Alencar. **Heurísticas e busca local estocástica para o sequenciamento just-in-time de máquinas paralelas sujeitas a janelas de tempo restritivas comuns**. 125 p. Dissertação de mestrado – Escola de Engenharia de São Carlos, Universidade de São Paulo, 2020.

Este estudo lida com problemas de sequenciamento com penalidades por atrasos e adiantamentos submetidos a prazos de entrega os quais podem ser representados por um instante ou intervalo de tempo. Problemas relacionados compartilham um forte viés prático, tendo em vista a às naturezas exógenas e endógenas das estruturas de custo que ocorrem em sistemas de produção *just-in-time*. Apesar da importância desses problemas, não há estudos recentes que abordam as suas duas principais variantes. Cobrimos essa lacuna através de uma revisão de mais de 200 artigos que representam mais de 40 anos de literatura. Também classificamos algoritmos para os casos onde os prazos (janelas) comuns de entrega são um parâmetro previamente estabelecido, assim como aqueles onde os prazos (janelas) são uma variável de decisão. Novas notações são introduzidas e um quadro de classificação que inclui uma vasta diversidade de problemas relacionados a atrasos e adiantamentos são introduzidos. Ademais, um total de 26 propriedades estruturais e a complexidade computacional de algoritmos para os casos de máquinas únicas, paralelas e *flow-shop* são sumarizadas. A partir da revisão de literatura, selecionamos um problema relativo ao sequenciamento *just-in-time* de máquinas paralelas submetidas a janelas de entrega restritivas comuns, o qual é conhecido por ser NP-difícil. Pelo fato de não existir procedimentos de resolução para problemas com mais de 40 tarefas, propomos uma família de heurísticas com desempenho promissor e capazes de resolver instâncias com até 500 tarefas. Além disso, a heurística com melhor desempenho é combinada com uma busca local estocástica (*Iterated Greedy*) que é capaz de melhorar ainda mais as soluções previamente encontradas.

Palavras-chave: Sequenciamento; Atraso e adiantamento; Janelas de tempo comuns; Máquinas paralelas; Heurísticas; Busca local estocástica.

List of Figures

List of Tables

List of Algorithms

Contents

Chapter 1

Introduction

Scheduling may be defined as the allocation of resources over time to perform tasks (MACCARTHY; LIU, 1993). Most of the scheduling problems are commonly found in industrial environments, which require operations to be performed in single or multiple machines to process production orders. For this reason, the importance of good scheduling strategies cannot be over-stressed. There is a clear and increasing necessity to respond to market demands quickly and to run plants efficiently, which leads to complex scheduling problems in all but the simplest production environments. It is worth noting that these problems are not exclusively found in industry, but in flight operations, execution of computer programs, and many others.

The scheduling function in a production facility or in a service organization must interact with many other functions. These interactions must satisfy system-dependent specifications and may take different forms. They usually occur within an enterprise-wide information system, which is often composed of a central computer and a database. There are also networks of personal computers, workstations, and data entry terminals, that are connected to the central computer with the objective of retrieving data from the database or entering new data. The software that controls such an elaborate information system is typically referred to as an Enterprise Resource Planning (ERP) system. Scheduling is frequently done interactively via a decision support system installed on a personal computer or workstation linked to the ERP system.

In this context, a schedule may be defined as the assignment of jobs over time onto machines and the scheduling problem is to optimize some predefined performance measure. In order to translate different production environments, it is necessary to establish the different technological constraints as well as the scheduling objectives. With that said, the class of scheduling problems studied in this master's thesis arise in two different application areas: production and computer engineering. Their common feature relies on the fact that it is crucial for the control system to observe the due dates.

Just-in-time (JIT) is a production planning and control (PPC) strategy that seeks to eliminate as much waste as possible. Completing a task before or after the due date incurs

additional cost, which is a waste of resources. Therefore, in JIT systems, it is desirable to complete a task as close to its due date as possible. In computer systems, the idea of observing due dates is typical for real-time systems. A system is said to be real-time if the correctness of an operation depends not only on its logical correctness but also on the time it is performed. As a matter of fact, it appears that the same type of algorithm may be applied for solving problems in both cases (JÓZEFOWSKA, 2007).

Despite the similarity of the above mentioned problems, the specific focus of this study is related to JIT scheduling in industrial environments. A section devoted to real-time systems is given to illustrate their practical applications. That being so, the rest of this chapter is divided as follows: first, JIT manufacturing is contextualized; second, the applications of JIT scheduling in real-time systems is carefully described; finally, the justification of the study is detailed according to the historical perspectives of the topic, practical importance, and literature gap.

1.1 Just-in-time manufacturing

Although a relatively recent concept, JIT philosophy is one of the key approaches to modern manufacturing planning and control. It represents a set of techniques that are employed in most managerial tasks, specially for PPC, which is responsible for determining the flow of materials through the manufacturing process. According to Józefowska (2007), the main goal of PPC is devising plans to balance the external demands of the marketplace with the resources and capacity of the manufacturing system. This goal is translated hierarchically from long to short range, at the following levels:

1. Strategic business planning.

2. Sales and operations planning.

3. Master production scheduling.

4. Material requirements planning.

5. Purchasing and production activity control.

Of course, each level may vary on its purpose, time span, and level of detail. Shifting from strategic plan to production activity control, the objective changes from general direction to specific detailed planning, the time span decreases from years to days, and the level of detail from general categories to individual components and machines.

First, the strategic business plan incorporates a major statement containing the objectives and directions the company expects to achieve in a time span usually ranging

from two to ten years. This plan is based upon a long-term forecast and includes participation of diverse areas of the organization such as marketing, finance, production, and engineering.

Once the major objectives are determined by the strategic plan, the sales and production plan is created. Commonly, the production management team decides the quantities of each product group that must be manufactured to meet external demands, the desired inventory levels, and the required resources. Such decisions must consider two important factors: work-in-process inventory and the scheduling horizon. The first one is related to products in various stages of completion through the manufacturing process, that is, from raw materials in early production stages to finished goods awaiting for the final inspection. The second one relates to the time period addressed by the plan.

Subsequently, the master production schedule is responsible for planning the production of finished goods. It breaks down the sales and operations to show, for each period, the quantities to be manufactured. Moreover, to satisfy the master production schedule, a material requirement plan should be created for establishing a purchase strategy capable of matching the amount of items necessary to manufacture a product with the stipulated demand. With that said, the material requirement plan should address two important factors:

1. The quantity of all items and materials required to produce the prescribed amount of products.

2. The due date in which these items are required.

Finally, purchasing and production activity control represents the implementation and control phase of the PPC system. Often, this activity has a high level of detail and a short planning horizon with the understanding that purchasing is responsible for establishing and controlling the flow of raw materials into the factory and production activity is conducted to manage the flow of work.

To get the most profit, a company must aim at providing the best costumer service with minimum production, distribution, and inventory costs. Obviously, providing good customer service conflicts with minimization of both production and inventory costs. The PPC systems that explicitly set goals to supply costumers in the exact time and to maintain inventory costs as low as possible are often called JIT systems.

1.2 Real-time systems

A real-time system may be defined as any information processing system that has to respond to externally generated stimuli within a specified and finite period. A correct performance of a task is associated not only with its logical result but also with the time

it is delivered. In this context, a failure to respond on time is as bad as giving an incorrect response (JÓZEFOWSKA, 2007).

Systems operating under time constraints occur within many application areas such as embedded systems, vehicle control, industrial plant automation, robotics, multimedia, audio and video stream conditioning, monitoring, and stock exchange orders. In more tangible terms, a good example is the keyboard and monitor of personal computers. An user must get visual feedback of each keystroke within a reasonable time, that is, if the user cannot see the output of the keystroke in the monitor within a certain time interval, the software product will be definitely strange to use. For instance, if the maximum acceptable period is 100 milliseconds, then any response between 0 and 100 milliseconds would be acceptable. Another example may be the anti-lock brakes in a car. The real-time constraint is the short time in which the breaks must be released to avoid the wheel from locking.

Real-time systems constitute an important part of contemporary computer science. Since real-time computing is ubiquitous, most of its applications remain unnoticed to the user. Actually, several devices are equipped with embedded systems, which are special-purpose systems in which the computer is completely encapsulated by the device it controls. In contrast with general-purpose computers, an embedded system performs one or a few predefined tasks, commonly with very specific requirements. Since the system is dedicated to specific tasks, design engineers can optimize it, reducing the size and the cost of the product.

As evidenced in the several examples given, it is possible to notice the similarities between problems in both manufacturing and computer systems. In the domain of computer systems, the idea of JIT scheduling is fundamental due to the fact that a task has to be completed within a time window; otherwise a system failure may incur.

1.3 Justification

In the past few decades, scheduling problems have received an important attention from operational research practitioners. However, for many years, much attention was devoted to regular performance measures, which are non-decreasing functions of job completion times. Such measures may include the number of tardy jobs, total tardiness, and total flow time. Notably, the first two criteria are the standard way for measuring the conformance to due dates, even though they neglect the costs that early jobs may incur (BAKER; SCUDDER, 1990). The emergence of JIT philosophy highlighted the importance of considering the cost structures associated with both early and tardy jobs and culminated in the rapid diffusion of non-regular performance measures.

In earliness and tardiness (ET) scheduling, the due dates may be treated as given constraints or decision variables. The context in which due dates are given in advance

reflects frequent situations where shop floor activities must meet customers demands, which, in turn, are influenced by external factors. On the other hand, treating them as decision variables may translate some production environments that set goals to guide the progress of internal tasks.

An example of real-world application of ET scheduling is seen in modern semiconductor fabrication facilities. The wafer fabrication part is characterized by several process restrictions which may culminate in common due date scheduling problems with ET oriented criteria, i.e., the situation found in memory business where it is likely that a large number of chips have the same external due date (loose or tight) and hence the same internal due date for the corresponding wafer fabrication facility. A second motivation is seen in base wafers, which are held on stock in a wafer bank for fulfilling specific customer requests that are dependent upon contractual obligations. In this situation, the due dates of almost all final chips are the same (MÖNCH; UNBEHAUN; CHOUNG, 2006; MÖNCH et al., 2011; ROCHOLL; MÖNCH, 2018).

Among the various ET models, this study target the ones where jobs share a common due date. These models coincide with assembly lines that require multiple components to build a final product or situations when several jobs compose a single customer order (SIDNEY, 1977; KANET, 1981). In addition, it is specially concerned with problems that allow a common tolerance interval (due window) in which no penalties incur. In many industries, it may be more suitable than considering a single point in time. Examples of practical applications of due windows include JIT manufacturing (WU; LAI, 2007), chemical processing and PERT/CPM scheduling (KOULAMAS, 1996) as well as information technology (JANIAK et al., 2015).

1.3.1 General objective

As discussed on the previous sections of this chapter, ET scheduling is present in many practical situations from industrial environments to real-time systems. For this reason, the first objective of this master's thesis is to provide an extensive literature review of ET scheduling against common due dates and windows. There are no other study or survey paper dealing with the cases where the due date is both a decision variable and given parameter. We fill this gap by proposing a new unified framework based on the due date approach, a complete summary of current literature, and the structural properties of related problems.

Based on the structural properties, the second general objective of this study is to develop new efficient heuristic methods for scheduling identical parallel machines against common restrictive due windows. An objective function with job-dependent ET weights ideally translates many practical practical contexts. The application of heuristic methods is justified due to the inefficiency of exact approaches in solving large-size problem

instances when the common due window is restrictive, for example, the best-so-far exact method is only capable of solving instances with up to 40 jobs in size (CHEN; LEE, 2002).

1.3.2 Specific objectives

The two general objectives can be subdivided in five specific goals that compose the motivation of this study:

1. Provide an extensive review of studies that deal with the ET objective functions in order to obtain a historical background of properties for the single, parallel and flow-shop machine environments.

2. Present the definition of the parallel-machine scheduling problem against common due windows and show a procedure to solve instances with up to 500 jobs and different restriction factors.

3. Develop constructive heuristics which are computationally efficient and easy to implement. This is critical for dealing with large-size problem instances in real-world applications.

4. Once an initial solution is obtained, a stochastic local search is applied to improve the results of the best-so-far constructive heuristic.

5. Compare the efficiency of the proposed algorithms in terms of solution quality and the computational effort to obtain the solutions.

The rest of this master's thesis is organized as follows: the second chapter is devoted for presenting the literature review on ET scheduling against common due dates and windows, a classification scheme for this class of problems, and their structural properties; the third chapter relates to the definition of the identical parallel-machine scheduling problem against common restrictive due windows; the fourth chapter describes the methodology adopted to tackle problem; the fifth chapter shows the results of the algorithms and their comparison; finally, the sixth chapter presents some concluding remarks.

Chapter 2

Literature review

This chapter presents an extensive literature review concerning ET scheduling against common due dates and windows and its historical development. It covers problems when the common due date is a variable to be determined or a given parameter. In the first case, the focus is directed to the constant flow allowance method (CON), which is one of the simplest models of due date assignment and consists of determining the position of a common due date. Other models encompass assignment policies that are dependent on job processing times, such as: slack due dates (SLK); total-work-content (TWK); processing-plus-wait due dates (PPW). There are also the ones that assign due dates to jobs based on a given set of values, such as generalized due dates (GDD) and unrestricted due date (where each job is assigned a different due date with no restrictions and also known as DIF). For survey papers dealing with different assignment models we refer to Gordon, Proth and Chu (2002a) and Gordon, Strusevich and Dolgui (2012). Note that, initial studies dealing with due date assignment relied on simulation techniques to find better due dates (EILON; CHOWDHURY, 1977; WEEKS; FRYER, 1977; SCULLI; TSANG, 1990). A survey addressing the evaluation of various dispatching rules in a dynamic context is due to Cheng and Gupta (1989). In a deterministic sense, important surveys include the ones by Baker and Scudder (1990) and Gordon, Proth and Chu (2002b). Here, a special attention is paid to problems in the deterministic case.

Despite the practical relevance of ET scheduling, there is no recent survey with an emphasis on common due dates and common due windows that holistically address them as both decision variables and given constraints. It is worth mentioning that these problems share inherent structural properties that must not be neglected. Baker and Scudder (1990) were the first to provide a general classification for ET problems, however, their survey only cover papers until 1990. Gordon, Proth and Chu (2002b) introduce a new nomenclature and an unified framework for common due date assignment problems, although, they do not include the ones where it is a given parameter. In addition, they focus specially on single and parallel machine environments. Lauff and Werner (2004b) review ET scheduling in multi-machine problems subject to a common due date, however they

only cover about 25 papers. More recently, Janiak et al. (2015) provide a general review on problems with due windows, but do not address the common due date case. Recently, Kramer and Subramanian (2019) present an annotated bibliography for problems involving single and parallel machines covering many objective functions with job-dependent weights, but without the specific focus on common due dates and windows. With that said, the contributions are presented:

- ❏ A new comprehensive literature review covering several structural properties established from early literature to present. In a similar fashion,Gordon, Proth and Chu (2002b) expanded the results in Baker and Scudder (1990) by including papers from 1990 to 2002, however, only for common due date assignment problems. Note that this study addresses the cases when the common due date is given in advance.

- ❏ A complete summary is given in the format of tables in each section/subsection. Special cases and conditions of the problems (such as agreeable weights, processing in batches, rate-modifying activities, machine availability,learning, ageing, deterioration, etc.) are also included.

- ❏ In addition to single and parallel machines, this study is the first to incorporate papers dealing with the flow-shop environment.

- ❏ A new unified framework based on the due date approach as well as the objective function for a general class of ET problems.

2.1 Scope, problem definition, and notations

The class of problems investigated is given as follows: there is a set of n available jobs $N = \{1, 2, ..., n\}$ that have to be processed on m $(M_1, M_2, ..., M_m)$ available machines. Each job $j \in N$ is characterized by a set of m processing times p_{ij} $(i = 1, 2, ..., m)$. All the jobs are subject to common due date d, which may be given in advance or may be treated as a decision variable. For problems where the common due date is a decision variable, the objective is to find the value of d together with a schedule π that minimizes an ET cost function. In a parallel machine context, each job consists of a single operation that has to be processed exactly on one of the m identical, uniform, or unrelated parallel machines. In the flow shop environment, each job has to be processed in series following the same given technological route. If the flow shop has k-stages with more than one machine, it is said to be a flexible (hybrid) flow shop. In single-machine problems, the machine index is dropped and the processing time of job j is denoted as p_j (also applies for identical parallel machines). There is also a given set of weights, e.g., α_j, β_j for each job. If C_j is the completion time of job j in a certain schedule, then the earliness (E_j)

and the tardiness (T_j) of job j are given by:

$$E_j = \max\{0, d - C_j\} \quad \text{and} \quad T_j = \max\{0, C_j - d\}$$

Some general problems may also include the number of early (V_j) and tardy jobs (U_j) that are calculated as:

$$V_j = \begin{cases} 1, & \text{if} \quad d - C_j > 0 \\ 0, & \text{otherwise} \end{cases} \quad \text{and} \quad U_j = \begin{cases} 1, & \text{if} \quad C_j - d > 0 \\ 0, & \text{otherwise} \end{cases}$$

In case of a common due window, it is assumed that no penalties incur between the earliest due date d_l (left boundary) and the latest due date d_r (right boundary) in such a way that earliness and tardiness penalties are now calculated as:

$$E_j = \max\{0, d_l - C_j\} \quad \text{and} \quad T_j = \max\{0, C_j - d_r\}$$

Since the vast majority of models considers a continuous penalty function when tolerance intervals are allowed, this configuration is implicitly assumed. Discontinuous penalty functions are mentioned when appropriate.

The well-known three field notation is used to describe the scheduling problems (GRAHAM et al., 1979). The first and the third fields designate the machine environment and the objective function, respectively. The second field is devoted to characterize special conditions of the problem. The following notations are used within the second field:

- ❏ $\langle d \rangle$: jobs are subject to a common due date.
- ❏ $\langle d_j \rangle$: jobs are subject to multiple common due dates.
- ❏ $\langle d^{\min} \rangle$: problems of finding the minimum due date among the optimal ones.
- ❏ $\langle d_l, d_r \rangle$: jobs are subject to a common due window.
- ❏ $\langle d(\pm u) \rangle$ due date with symmetric tolerance interval u.
- ❏ $\langle d - u, d + v \rangle$ due date with asymmetric tolerance interval.
- ❏ $\langle d \pm (u_j) \rangle$ jobs have tolerances of different sizes.

The notation $\langle \hat{d} \rangle$ is introduced to designate problems where the due date is restricted by $d < p_1 + p_2 + ... + p_n$. As we shall discuss later, optimal solutions for these problems share different structures in comparison to the ones where the due date is unrestrictedly large. This is also valid for problems with a common due window with the understanding that its left boundary is restricted. To denote restricted due windows, the notation $\langle \hat{d}_l, \hat{d}_r \rangle$ is adopted. Other important notations in the second field are given as follows:

- ❏ **B**: Batch formation process.

- f: job families.
- ST_{sd}: sequence-dependent set-up times.
- $ST_{si,b}$: sequence-independent batch or family set-up.
- $ST_{sd,b}$: sequence-dependent batch or family set-up.
- rm: rate-modifying activity.
- mrm: multiple rate-modifying activities.
- r_j: nonzero release dates.
- agr: agreeable rates condition.
- prp: proportional weights condition.
- $nonprmu$: nonpermutation flow shop.
- $no-wait$: no-wait flow shop.
- $size$: multiprocessor tasks.
- dtr: deterioration effects.
- lrn: learning effects.

If batching is allowed, it is distinguished between problems where the batches are given parameters (already partitioned into f, $f \geq 1$, families) and the ones where the batch formation is part of the decision process (denoted as **B**). For classifying the set-up times, we use the notation in Allahverdi et al. (2008). The set-ups are sequence-dependent if its duration depends on the families of both the current and the immediately preceding batches. It is said to be sequence-independent if its duration depends solely on the family of the current batch. In non-batching environments, the concepts of sequence-dependent and sequence-independent set-ups hold with the understanding that each family is composed by a single job.

Some models also allow a rate-modifying (rm) activity to be performed. When the machine is undergoing rm, no production is possible. The processing time of job j remains p_j if it is processed prior to rm and $\psi_j p_j$ if it is scheduled after it, with ψ_j denoting the modifying rate (usually $0 < \psi_j \leq 1$). Similarly, problems with learning effects also permit reducing the processing time to a certain extent, however, since there are many different equations used to model these effects, it was decided to use the notation lrn and mention the specific characteristics of each approach throughout the text. It is worth noting that some authors consider that the processing times are controllable, that is, it is possible to reduce the processing time of a job j by allocating a limited amount of resource x_j to it.

If λ_j is the unitary cost, then $\lambda_j(x_j)$ is the processing cost of job j. Since the reduction of the compression costs is commonly one of the objective-function components, the notation $\lambda_j(x_j)$ is incorporated into the third field and the functions that determine the resource allocation are mentioned when appropriate.

In contrast with the approaches that permit the reduction of processing times, some authors assume that the processing times increase according to its starting time or position in sequence. In these cases, the jobs are said to be under the effects of ageing or deterioration. To illustrate a simple deterioration effect that depends upon the starting time of a job, let p_j denote the actual processing time of job j. A linear deterioration may be given as $p_j = \hat{p}_j + \omega \times S_j$, where $\hat{p}_j$ is the normal processing time, ω is a given constant, and S_j is the starting time of job j. Of course, there are many other ways to express these effects. For this reason, it was decided to use the notation *dtr* in the second field and mention the particularities separately. Furthermore, if the duration a rate-modifying activity is subject to the effects of deterioration, we specify it in the second field using $dtr - rm$.

In some special cases, it is assumed that the weights associated with each job are subject to certain restrictions. For instance, if the ET weights for any two jobs j and k are such that $p_j/\alpha_j < p_k/\alpha_k$ implies $p_j/\beta_j \leq p_k/\beta_k$, then it is said that the weights are agreeable (*agr*). Other restriction is to set weights to be proportional (*prp*) to processing times, e.g., $\alpha_j = wp_j$, when w is a given constant.

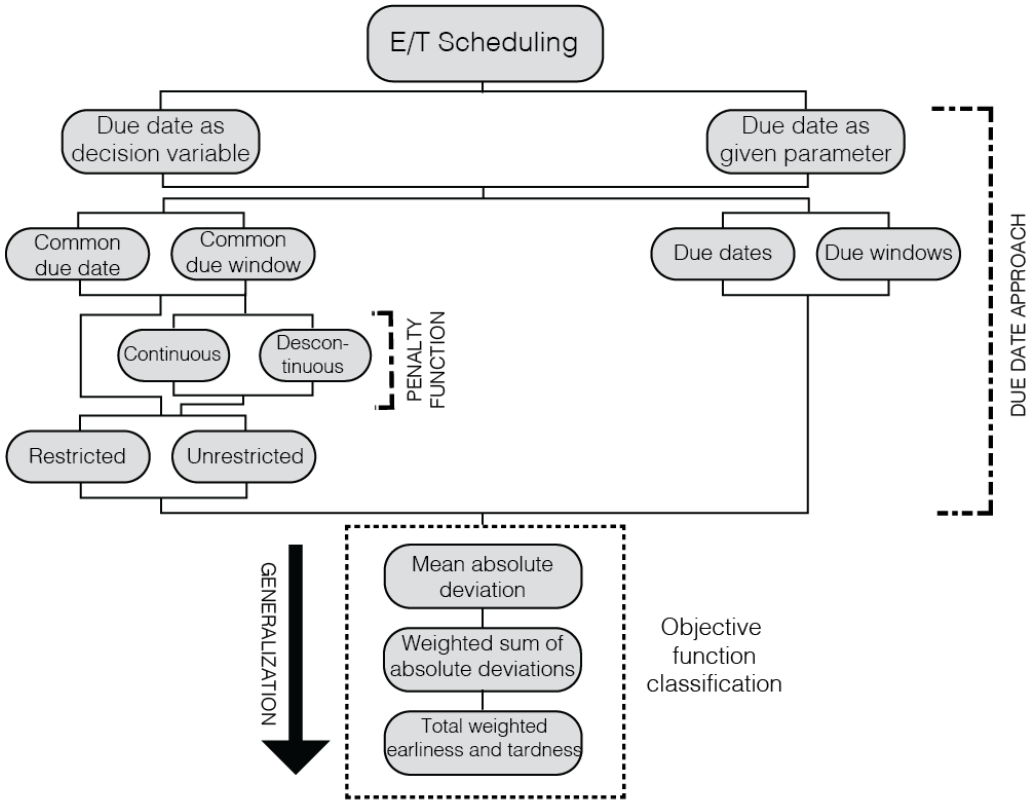
In flow shop problems, it is important to differentiate between permutation and non-permutation (*nonprmu*) schedules. If the job sequence is the same for all machines, the schedule is called permutational. On the other hand, if the sequence is allowed to change from one machine to the next, the problem is said to be *nonprmu*. Since the vast majority of papers considers only permutation schedules, this environment is implicitly assumed to be the standard. Non-permutation flow shops are mentioned when appropriate. The notation $no-wait$ is used to designate flow shops that do not allow waiting time between successive operations, that is, once the processing of a certain job is started, no interruptions are permitted between its processing in the next machines. Moreover, in case of flexible flow shops, the notation *size* indicates that one job can simultaneously visit $size_{ij}$ out of m_i machines on a given stage.

For classifying solution approaches, the notations *DP* and *Enumer.* are adopted to refer, respectively, to exact methods based on dynamic programming (commonly pseudopolynomial time algorithms) and branch-and-bound procedures or mathematical programming formulations (the ones that involve a partial enumeration of all sets of feasible solutions); *Approx.* and *Heuristic* denote algorithms that do not ensure optimality, however, may allow finding optimal solutions, respectively, with or without an estimation on how these solutions differ from the optimal ones (MACCARTHY; LIU, 1993; GORDON; PROTH; CHU, 2002b).

In terms of the notation used to classify objective functions based on ET penalties, the nomenclature in Gordon, Proth and Chu (2002b) is adopted. There exists at least three main variants of the problem, namely, the mean absolute deviation (MAD) from the common due date (penalties are equally assessed); weighted sum of absolute deviations (WSAD), which allow job-independent cost coefficients; total weighted earliness and tardiness (TWET), which can be further divided into problems with symmetric and asymmetric job-dependent weights. Moreover, these objective functions can be distinguished into the ones that seek to minimize the sum (*minsum*) or the maximum (*minmax*) value of penalties. Our primary focus are papers dealing with *minsum*. A framework depicting the classification of ET problems based on the due date approach as well as the objective functions is given in Fig. 1.

A framework containing the structure of E/T problems is given in Figure 1.

Figure 1 – Framework of E/T scheduling problems



2.2 Single-machine models

The vast majority of ET models has been developed for the single-machine case. This includes several inherent properties of optimal solutions. Throughout this section, we will discuss the classic problem, which may be constituted by the MAD, WSAD, and TWET cost functions subject to a common due date. Later, we mention the cases when jobs are

subject to a common due window and its variations. Finally, we consider problems with batching and job families, learning effects, and nonlinear cost-functions, respectively.

Hereafter, at the end of each subsection, we provide summary tables, which are organized as follows. The column *problem* classifies the problems according to the notations given in Section 2.1. Problems that are solved in polynomial time are given first followed by NP-hard problems. Finally, the column *reference* associates the type of problem with the solution procedure given in column *algorithm*. It is worth noting that, in many cases, there are no polynomial-time algorithms for a given problem, however, some authors show special cases that permit such procedures. These cases are also listed in the column *algorithm*.

2.2.1 Mean absolute deviation

The simplest way of analysing ET problems is to set an objective function that measures the mean absolute deviation (MAD) around a common due date, which aim at finding a schedule and a due due date that minimizes:

$$f(\pi, d) = (1/n) \sum_{j=1}^{n} (E_j + T_j) \tag{1}$$

Without loss of generality, the constant $1/n$ is omitted and the objective function reduces to $\sum_{j=1}^{n} |C_j - d|$. Since jobs are penalized at the same rate, the objective is to build a schedule so that the jobs are equally assigned before and after d. It is worth noting that, if d is too tight, it may not be possible to fit enough jobs in front of d and the problem may become restricted.

In this section, we assume that jobs are arranged in non-decreasing order of processing times with $\pi = ([1], [2], ..., [n])$ where $[j]$ denotes jth job in the sequence. Let B be the set of jobs that complete on or before d and A be the set of jobs that complete after d. Then, the ordered sets B and A are defined as $B = \{n, n-2, ..., 1\}$, $A = \{2, 4, ..., n-1\}$ when n is odd, and $B = \{n, n-2, ..., 2\}$, $A = \{1, 3, ..., n-1\}$ when n is even. Now, let $b = |B|$ and $a = |A|$ be the number of jobs in each set. From the definitions of sets A and B, $b = a$ if n is even and $b = a + 1$ otherwise. Let $\Delta = \sum_{j \in B}$. With that said, an optimal schedule may be characterized by the following properties (BAKER; SCUDDER, 1990):

- **Property 1**: There is no inserted idle time in a schedule.
- **Property 2**: An optimal schedule is V-shaped. Jobs with $C_j \leq d$ are sequenced by long processing time first (LPT) and jobs with $C_j > d$ are sequenced by shortest processing time first (SPT).
- **Property 3**: One job finishes processing exactly at d ($C_j = d$ for some j).

- **Property 4**: In an optimal schedule, the bth job in sequence completes precisely at time d, where b is the smallest integer greater than or equal to $n/2$ if n is even and $(n+1)/2$, otherwise.

With the understanding that $B \cup A$ gives an optimal schedule, Kanet (1981) provides an efficient algorithm that requires $d \geq \sum_{j=1}^{n} p_j$ to ensure that the schedule starts after time zero. Bagchi, Sullivan and Chang (1986) shows that there are at least $2^{n/2}$ optimal schedules if n is even and $2^{(n+1)/2}$ if n is odd and show an algorithm for finding all optimal solutions. Furthermore, they also argue that the problem remains unrestricted if $d \geq \Delta$.

For the restricted version ($d < \Delta$) properties 1 and 2 still hold. The proof of property 2 for the constrained problem is found in Raghavachari (1986). However, optimal solutions for this type of problem may contain a straddling job. In other words, one that starts before the common due date and completes after it. Then, property 3 and 4 do not hold. Bagchi, Sullivan and Chang (1986) outline an enumeration algorithm for the restricted problem and implicitly assumed that the start time of the schedule is zero. Similarly, Sundararaghavan and Ahmed (1984) outline a heuristic procedure. Although, Szwarc (1989) argues that an optimal schedule may not start at time zero when d is restricted. Thus, the enumeration approach of Bagchi, Sullivan and Chang (1986) may not guarantee optimal solutions.

To deal with the issue of ignoring delayed starts, Hall, Kubiak and Sethi (1991) rely o a dynamic programming algorithm called Optcet, which includes subroutines EVS, TVS, and Nosplit and runs in $O(n \sum p_j)$. The authors report that this algorithm produces optimal solutions even for instances with up to 1000 jobs. They prove that the restricted problem is NP-complete in the ordinary sense and deduce the following property:

- **Property 5**: In an optimal schedule, there exists a job that either starts at time zero or completes precisely at the due date.

Ventura and Weng (1995) claim that subroutines EVS and TVS can be eliminated and, consequently, the total computational effort can be further reduced. Hoogeveen, Oosterhout and Velde (1994) introduce a branch-and-bound algorithm based on Lagrangean lower and upper bounds that is effective at producing optimal solutions to instances with up to 50 jobs. They also show a polynomial time 4/3-approximation algorithm that runs in $O(n \log n)$ and guarantees solutions with values at most 4/3 times the optimal solution.

In the context of sequence-dependent set-up times, if d remains unrestricted, property 1 still holds, simply due to the fact that removing any gaps in the schedule would make the completion times closer to the due date. Properties 3 and 4 also holds, however, the optimal schedule may not be V-shaped. Hence, property 2 does not hold (RABADI; MOLLAGHASEMI; ANAGNOSTOPOULOS, 2004). If sequence-dependent set-up times are included, there is no available efficient procedure available. Not even pseudopolynomial

time dynamic programming approaches are found in literature. Rabadi, Mollaghasemi and Anagnostopoulos (2004) rely on a branch-and-bound algorithm, but their solution approach only solves instances with up to 25 jobs in size. For larger instances, Rabadi, Anagnostopoulos and Mollaghasemi (2007) propose a shortest adjusted processing time first (SAPT) heuristic, which is composed of a construction and a local search phase. The SAPT heuristic runs in $O(n^2)$. To refine solutions, they incorporate a simulated annealing (SA) into the SAPT. More lately, Hepdogan et al. (2009) proposed a metaheuristic for randomized priority search (Meta-RaPS), which is a generic, high level strategy used to modify greedy algorithms based on the insertion of a random element. They combine the metaheuristic with the SAPT heuristic to obtain results that outperform the integration of SAPT and SA described in Rabadi, Anagnostopoulos and Mollaghasemi (2007).

Table 1 – Summary of algorithms for the single-machine MAD problem

Problem	Complexity	Reference	Algorithm
$1\|\langle d\rangle\|\sum(E_j+T_j)$	P	Kanet (1981)	$O(n\log n)$
		Panwalkar, Smith and Seidmann (1982)	$O(n\log n)$
		Hall (1986)	$O(n\log n)$
		Bagchi, Sullivan and Chang (1986)	$O(n\log n)$
		Bagchi, Chang and Sullivan (1987)	$O(n\log n)$
$1\|\langle d_j\rangle\|\sum(E_j+T_j)$	P	Dickman, Wilamowsky and Epstein (2001)	$O(n\log n)$
$1\|\langle\hat{d}\rangle\|\sum(E_j+T_j)$	NP-hard	Sundararaghavan and Ahmed (1984)	*Heuristic*
		Bagchi, Sullivan and Chang (1986)	*Enumer.*
		Szwarc (1989)	*Enumer.*
		Hoogeveen, Oosterhout and Velde (1994)	*Enumer.*
		Hoogeveen, Oosterhout and Velde (1994)	*Approx.*
		Hall and Posner (1991)	*DP*
		De, Ghosh and Wells (1993)	*DP*
		Ventura and Weng (1995)	*DP*
$1\|\langle d\rangle, ST_{sd}\|\sum(E_j+T_j)$	NP-hard	Rabadi, Mollaghasemi and Anagnostopoulos (2004)	*Enumer.*
		Rabadi, Anagnostopoulos and Mollaghasemi (2007)	*Heuristic*
		Hepdogan et al. (2009)	*Heuristic*

Dickman, Wilamowsky and Epstein (2001) introduce the problem with multiple common due dates to minimize MAD. Now, there is a set of y common due dates D'_i.

The problem is to find a schedule together with common due dates D'_i to minimize $f(\pi, D'_i) = \sum_{j=1}^{n} X_j$, where $X_j = \min(|C_j - D'_1| + |C_j - D'_2|, +... + |C_j - D'_y|)$. They show that this problem is equivalent to multi-machine problem in Sundararaghavan and Ahmed (1984), therefore, a similar solution procedure can be applied to obtain optimal solutions. Table 1 gives the summary of algorithms for single-machine problems with a MAD objective function.

2.2.2 Weighted sum of absolute deviations

In contrast with the MAD problem, the weighted sum of absolute deviations (WSAD) problem allows a different cost for earliness and tardiness. A more general objective function is introduced in Panwalkar, Smith and Seidmann (1982), when a due date penalty is also considered. This structure makes practical sense if it is taken into account that a company might offer price discounts while negotiating with a client in case of large due dates. It is represented as follows:

$$f(\pi, d) = \sum_{j=1}^{n} (\alpha E_j + \beta T_j + \gamma d) \tag{2}$$

Properties 1, 2, and 3 hold if $\gamma < \beta$ and property 4 takes the following form (BAKER; SCUDDER, 1990):

- ❑ **Property 6**: In an optimal schedule, the bth job in sequence completes precisely at time d, where b is the smallest integer greater than or equal to $n(\beta - \gamma)/(\alpha + \beta)$.

If $\gamma \geq \beta$ the interpretation is that the SPT order is optimal and the due date should be set to zero, therefore $B \to \varnothing$. Panwalkar, Smith and Seidmann (1982) propose an algorithm that runs in $O(n \log n)$ that yields an optimal solution. Cheng (1988a) points out that if $b = n(\beta - \gamma)/(\alpha + \beta)$ it is not necessary to set $C_{[b]} = d$ to obtain an optimal solution. Rather, an optimal due date may be set anywhere in the range $C_{[b]} + \epsilon p_{[b+1]}$, $0 \leq \epsilon < 1$. Consequently, there exists an infinite number of optimal due dates, for example, if n is even, $\gamma = 0$, and $\alpha = \beta$, then $b = n/2$ and any value between $C_{[n/2]}$ and $C_{[(n/2)+1]}$ may lead to an optimal due date.

An efficient procedure that runs in $O(n \log n)$ that produces an optimal schedule and the minimum value of d for the unrestricted problem is given by Bagchi, Chang and Sullivan (1987). Similarly to the MAD problem, Bagchi, Chang and Sullivan (1987) provide an enumeration scheme that solves the restricted version, however, also imposes that the start time of a schedule should be zero.

For solving the restricted version of the problem optimally (without the assumption that a schedule starts at time zero), Mondal and Sen (2001) provide a branch-and-bound algorithm that uses a graph search space. New upper and lower bounds are proposed for instances with up to 2000 jobs.

If jobs are non-simultaneously available, each job is assigned a release date. Now, let r_j refer to the release date of job j. Cheng, Chen and Shakhlevich (2002) were the first to prove that the due date assignment problem and scheduling is strongly NP-hard. They show solution procedures when either the job sequence is predetermined or all jobs share the same processing times. Approximation algorithms are also shown for the general problem as well as for some special cases. More recently, Birgin and Ronconi (2012) extend the results in Sridharan and Zhou (1996) and provide a heuristic procedure for the WSAD problem with ready times when the due dates are restricted and given in advance. Their algorithm is composed of a construction phase and two improvement phases with the overall complexity estimated in $O(n^2)$.

Kim and Lee (2009) considers the WSAD due date assignment problem when sequence-dependent set-up times occur. They first propose a branch-and-bound algorithm that solves small instances (up to 16 jobs). For larger instances, a two-phase heuristic procedure yields close to optimal solutions when compared with the upper bounds established by the exact method.

Panwalkar and Rajagopalan (1992) report the case when the processing time of a job can be reduced within a certain extent. In other words, the processing time of any given job can be chosen within a time interval that is previously stipulated. The processing time p_j of job j may be chosen in the interval $[p''_j, p'_j]$, where $p_j = p'_j$ is the normal time, which can be reduced as long as it remains greater than p''_j. The processing time can be reduced at an unitary cost λ_j. If x_j is the the amount of time compressed, then, $\lambda_j(x_j)$ the processing cost of j. Panwalkar and Rajagopalan (1992) provide an $O(n^3)$ algorithm and show that there is an optimal sequence without partial compression. Cheng, Oguz and Qi (1996) adds a due date penalty to the objective function and propose an efficient algorithm that runs in $O(n \log n)$ for the special case where all the jobs have a common upper bound for compression and an equal unit compression cost. Biskup and Cheng (1999b) includes a completion time penalty θ and conclude that the solution can also be found in $O(n^3)$ by solving an assignment problem. Biskup and Jahnke (2001) considers the case when all processing times should be reduced by the same cost $g(x)$, which is a monotonous increasing function in the proportion of the time reduction x. Again, the solution procedure runs in $O(n \log n)$. If $g(x)$ is a non-monotonous function, Ng et al. (2003) show that the optimal solution can be found in $O(n^2 \log n)$.

Leyvand, Shabtay and Steiner (2010) extend previous results of controllable processing times (PANWALKAR; RAJAGOPALAN, 1992; CHENG; OGUZ; QI, 1996; BISKUP; CHENG, 1999b) to cases when the time reduction is given by a generalized convex function of the amount of resource allocated to a job. The described solution procedure runs in $O(n^3)$. More recently, Koulamas (2017) provides a more generalized objective function including the weighted number of early and tardy jobs. The problem is solvable in $O(n^3)$ and $O(n^4)$ in case of the inclusion of controllable processing times with the convex resource

consumption function described in Leyvand, Shabtay and Steiner (2010). Table 2 gives the summary of algorithms for single-machine problems with a WSAD objective function.

2.2.3 Total weighted earliness and tardiness

Some models consider the fact that a particular job may have different ET penalties. Even if the weights are symmetric ($\alpha_j = \beta_j$), each job may have its own penalty. In general, this objective function is referred to as total weighted earliness and tardiness (TWET) and is summarized as:

$$f(\pi, d) = \sum_{j=1}^{n} (\alpha_j E_j + \beta_j T_j + \gamma d) \tag{3}$$

Similarly to the MAD problem, properties 1 and 3 hold and properties 2 and 4 take the more general form (QUADDUS, 1987; BAKER; SCUDDER, 1989):

- **Property 7**: An optimal schedule is V-shaped. Jobs with $C_j \leq d$ are sequenced in the non-increasing order of the ratio p_j/α_j and jobs with $C_j > d$ are sequenced in non-decreasing order of the ratio p_j/β_j.

- **Property 8**: In an optimal schedule, the bth job in sequence completes precisely at time d, where b is the smallest integer satisfying the inequality $\sum_{j=1}^{b} (\alpha_j + \beta_j) \geq \sum_{j=1}^{n} (\beta_j - \gamma)$

The variations of properties 7 and 8 for more specific cases are found in Cheng (1985), Cheng (1987), Bector, Gupta and Gupta (1988) and Hall and Posner (1991).

For solving the problem with symmetric weights and a large common due date, De, Ghosh and Wells (1990a) show a 0-1 quadratic programming formulation, a branch-and-bound scheme as well as two dynamic programming algorithms that run in $O(n \sum p_j)$ or $O(n \sum \alpha_j)$. Hall and Posner (1991) prove that even the problem with symmetric weights is NP-hard. They give a dynamic programming algorithm and an approximation scheme when the maximum weight is bounded by a polynomial function of n. Although, when this condition holds, Jurisch, Kubiak and Jozefowska (1997) show that problem is polynomially solvable. Contrarily to the assumption of a maximum bounded weight, Kovalyov and Kubiak (1999) give a fully polynomial-time approximation (FPTAS) scheme, which was the improved by Kellerer, Rustogi and Strusevich (2020). Cheng (1990) provide an $O(n^{1/2}2^n)$ partial search algorithm for finding an optimal sequence and due date, which represents a significant improvement in relation to the one in Cheng (1987). Hoogeveen and Velde (1991) present a pseudopolynomial algorithm that runs in $O(n^2 \sum p_j)$ for the restricted version with symmetric weights. For solving larger instances, Hao et al. (1996) propose a tabu search procedure. Kellerer and Strusevich (2010) show a FPTAS based on a special version of the knapsack problem to minimize a convex quadratic non-separable

Table 2 – Summary of algorithms for the single-machine WSAD problem

Problem	Complexity	Reference	Algorithm
$1\|\langle d\rangle\|\sum(\alpha E_j+\beta T_j)$	P	Panwalkar, Smith and Seidmann (1982)	$O(n\log n)$
		Bagchi, Chang and Sullivan (1987)	$O(n\log n)$
$1\|\langle d\rangle\|\sum(\alpha E_j+\beta T_j+\gamma d)$	P	Panwalkar, Smith and Seidmann (1982)	$O(n\log n)$
$1\|\langle d\rangle\|\sum(\alpha E_j+\beta T_j)+\lambda_j(x_j)$	P	Panwalkar and Rajagopalan (1992)	$O(n^3)$
		Leyvand, Shabtay and Steiner (2010)	$O(n^3)$
$1\|\langle d\rangle\|\sum(\alpha E_j+\beta T_j+\gamma d)+\lambda_j(x_j)$	P	Cheng, Oguz and Qi (1996)	$O(n^3)$
			$O(n\log n)$
		Leyvand, Shabtay and Steiner (2010)	$O(n^3)$
$1\|\langle d\rangle\|\sum(\alpha E_j+\beta T_j+\theta C_j)+\lambda_j(x_j)$	P	Biskup and Cheng (1999b)	
		Leyvand, Shabtay and Steiner (2010)	$O(n^3)$
$1\|\langle d\rangle\|\sum(\alpha E_j+\beta T_j+\gamma d)+g(x)$	P	Biskup and Jahnke (2001)	$O(n\log n)$
		Ng et al. (2003)	$O(n^2\log n)$
$1\|\langle d\rangle\|\sum(\alpha E_j+\beta T_j+\alpha' V_j+\beta' U_j)+\lambda_j(x_j)$	P	Koulamas (2017)	$O(n^4)$
		Koulamas (2017)	$O(n^3)$
$1\|\langle \hat{d}\rangle\|\sum(\alpha E_j+\beta T_j)$	NP-hard	Bagchi, Chang and Sullivan (1987)	*Enumer.*
		De, Ghosh and Wells (1993)	*DP*
		Mondal and Sen (2001)	*Enumer.*
$1\|\langle d\rangle, r_j\|\sum(\alpha E_j+\beta T_j+\gamma d)$	NP-hard	Cheng, Chen and Shakhlevich (2002)	*Heuristic*
$1\|\langle \hat{d}\rangle, r_j\|\sum(\alpha E_j+\beta T_j)$	NP-hard	Birgin and Ronconi (2012)	*Heuristic*
$1\|\langle d\rangle, ST_{sd}\|\sum(\alpha E_j+\beta T_j+\gamma d)$	NP-hard	Kim and Lee (2009)	*Enumer.*
			Heuristic

function. For the continuous relaxation of such knapsack problem, they give an algorithm of a quadratic time complexity. More recently, T'kindt, Shang and Croce (2019) proposed a sort and search algorithm for the unrestricted version of the problem with worst case time and space complexity of $O(2^{n/2})$.

Table 3 – Summary of algorithms for the single-machine TWET problem with polynomially solvable cases

Problem	Complexity	Reference	Algorithm
$1\|\langle d\rangle, prp\|\sum \alpha_j(E_j+T_j)$	P	Ahmed and Sundararaghavan (1990)	$O(n\log n)$
$1\|\langle \hat{d}\rangle, prp\|\sum \alpha_j(E_j+T_j)$	P	Alidaee and Dragan (1997)	$O(n\log n)$
$1\|\langle d\rangle\|\sum w_j(E_j+T_j)+\lambda(x)$	P	Wang (2006)	$O(n\log n)$
$1\|\langle d\rangle\|\sum w_j(E_j+T_j)+\lambda_j(x_j)$	P	Wang (2006)	$O(n^3)$

Under the agreeable rates condition, Ahmed and Sundararaghavan (1990) show that the unrestricted variant with symmetric weights is polynomially solvable. Later, Alidaee and Dragan (1997) extend the results for the restrictive case and show that the LPT order is optimal. Lee, Danusaputro and Lin (1991) impose the agreeable condition, however, add a penalty for tardy jobs in the objective function. They show dynamic programming procedures that run in $O(n\sum p_j)$ and $O(n\sum p_j+n^2d)$ for the due date as decision variable and given parameter, respectively. Later, Szwarc (1996) proposes several improvements.

Now, considering the unrestricted cases when $\alpha_j \neq \beta_j$, Dileepan (1993) proposes both heuristic and branching procedures. De, Ghosh and Wells (1994b) also rely on branching and heuristic methods, however, they add a due date penalty in the objective function. A heuristic approach is reported in Gupta, Bector and Gupta (1990) who also show solution procedures for several due date assignment methods. To find optimal solutions, Akker, Hoogeveen and Velde (2002) combine column generation and Lagrangean relaxation to speed up a branch-and-price algorithm. This technique allows to solve problems with up to 125 jobs, while pure column generation can solve instances with only 60 jobs.

The vast majority of restricted TWET problems were tackled by means of metaheuristic approaches. Lee and Kim (1995) were the first to propose genetic algorithms while James (1997) adopts tabu search. However, both neglect the fact that optimal schedules may have their starting times different than zero, which excludes possible optimal solutions from their search.

A methodology for developing instances with different due date tightness factors is provided in Biskup and Feldmann (2001). These factors are an adjustable parameter of a problem generator that contributes with benchmarks and permits to select how restrictive a due date should be. The authors also develop some heuristic procedures to solve 280 instances with more or less restricted due dates. Later, Feldmann and Biskup (2003) use the more restricted benchmarks to test the efficiency of their metaheuristics obtaining a an average improvement of 5%. Note that properties 1, 5, and 7 hold for the restricted

TWET. After the publication of these results, several papers focused on finding new upper bounds and designing more efficient methods.

Hino, Ronconi and Mendes (2005) propose a constructive heuristic called HRM to obtain an initial solution, two metaheuristics, and two hybrid strategies. Their results showed better solutions in 120 out of 140 benchmarks reported in Feldmann and Biskup (2003). Lin, Chou and Chen (2007) use genetic algorithms and simulated annealing with greedy local search which find slightly superior solutions (ranging from 0.19% to 5.84% depending on the due date factor) in relation to the two previous benchmarks. Lin, Chou and Ying (2007) show a sequential exchange approach, which is a heuristic method that yields, in many cases, better results when compared to the metaheuristics proposed in Feldmann and Biskup (2003) and requires less computational effort in relation to the ones in Hino, Ronconi and Mendes (2005). For the same problem and using the same benchmarks, Liao and Cheng (2007) propose a hybrid metaheuristic that uses tabu search within variable neighbourhood search. Despite finding much better solutions than the ones reported in Feldmann and Biskup (2003), but do not evaluate the results in relation to Hino, Ronconi and Mendes (2005). Ying (2008) incorporates aspects of the sequential exchange approach into a recovering beam search heuristic, which does not depend on parameter settings and other stochastic factors. Again, the previous benchmarks were outperformed (FELDMANN; BISKUP, 2003; HINO; RONCONI; MENDES, 2005). Nearchou (2008) rely on a differential evolution (DE) approach and find new upper bounds for 60% of the instances in Feldmann and Biskup (2003), specially for larger and more restrictive cases. An important contribution is also seen in Weng and Fujimura (2008), where they show a modified version of the HRM heuristic, which helps in achieving the fast contingency of the algorithm. They also propose a self-evolution algorithm, which performs faster in comparison to the metaheuristics in Hino, Ronconi and Mendes (2005). More recently, Yousefi and Yusuff (2013) adapts an imperialist competitive algorithm for discrete optimization and show that this method requires less computational effort in relation to the methods in Hino, Ronconi and Mendes (2005) and the DE of Nearchou (2008) specially for larger instances. Yuce et al. (2017) compare the effectiveness of a hybrid genetic bees algorithms with the algorithms presented in Feldmann and Biskup (2003), Hino, Ronconi and Mendes (2005), Lin, Chou and Ying (2007), and Nearchou (2008). Better results are achieved for more restricted large instances.

Still considering the restricted TWET problem with a given due date, Ronconi and Kawamura (2010) propose a branch-and-bound procedure, which is capable of solving instances with up to 35 jobs to optimality. Alvarez-Valdes et al. (2012) show two quadratic integer programming models and a heuristic that simply imposes a time limit to these models. They argue that optimality could be proven for all 10 and 20 job instances and nearly half of them for 50 to 500 using the benchmarks in Biskup and Feldmann (2001). For instances with up to 1000 jobs they could only guarantee optimality for few instances,

Table 4 – Summary of algorithms for NP-hard single-machine TWET problems

Problem	Complexity	Reference	Algorithm
$1\|\langle d\rangle\|\sum \alpha_j(E_j+T_j)$	NP-hard	De, Ghosh and Wells (1990a)	*Enumer.*
			DP
		Cheng (1990)	*Enumer.*
		Hall and Posner (1991)	$O(n)$
			$O(n \log n)$
			DP
		Hao et al. (1996)	*Heuristic*
		Jurisch, Kubiak and Jozefowska (1997)	*DP*
		Kovalyov and Kubiak (1999)	*Approx.*
		Kellerer, Rustogi and Strusevich (2020)	*Approx.*
		T'kindt, Shang and Croce (2019)	*Enumer.*
$1\|\langle \hat{d}\rangle\|\sum \alpha_j(E_j+T_j)$	NP-hard	Hoogeveen and Velde (1991)	*DP*
		Kellerer and Strusevich (2010)	*Approx.*
$1\|\langle d\rangle\|\sum (\alpha_j E_j+\beta_j T_j)$	NP-hard	Dileepan (1993)	*Enumer.*
			Heuristic
		De, Ghosh and Wells (1994b)	*Enumer.*
			Heuristic
		Akker, Hoogeveen and Velde (2002)	*Enumer.*
$1\|\langle \hat{d}\rangle\|\sum (\alpha_j E_j+\beta_j T_j)$	NP-hard	Lee and Kim (1995)	*Heuristic*
		James (1997)	*Heuristic*
		Biskup and Feldmann (2001)	*Heuristic*
		Feldmann and Biskup (2003)	*Heuristic*
		Hino, Ronconi and Mendes (2005)	*Heuristic*
		Lin, Chou and Chen (2007)	*Heuristic*
		Lin, Chou and Ying (2007)	*Heuristic*
		Ying (2008)	*Heuristic*
		Nearchou (2008)	*Heuristic*
		Weng and Fujimura (2008)	*Heuristic*
		Ronconi and Kawamura (2010)	*Enumer.*
		Alvarez-Valdes et al. (2012)	*Enumer.*
		Yousefi and Yusuff (2013)	*Heuristic*
		Yuce et al. (2017)	*Heuristic*
$1\|\langle \hat{d}\rangle\|\sum (\alpha_j E_j+\beta_j T_j)/C_j$	NP-hard	Giannopoulos and Nearchou (2018)	
$1\|\langle \hat{d}\rangle\|\sum (\alpha_j E_j+\beta_j T_j)/U_j$	NP-hard	Chen and Sheen (2007)	*Enumer.*
$1\|\langle d\rangle, agr\|\sum (\alpha_j E_j+\beta_j T_j+\alpha'_j U_j)$	NP-hard	Lee, Danusaputro and Lin (1991)	*DP*
			$O(n^2)$
$1\|\langle \hat{d}\rangle, agr\|\sum (\alpha_j E_j+\beta_j T_j+\alpha'_j U_j)$	NP-hard	Lee, Danusaputro and Lin (1991)	*DP*

hence, the use of heuristic approaches is recommended.

A further extension of the TWET problem is the inclusion of controllable processing times. Wang (2006) presents a model for determining a due date and a schedule so as to minimize penalties associated with the common due date, processing time reduction as well as the deviations from the due date, which has an unusual assumption that the weights are associated to the position in which the jobs are scheduled, rather than a fixed value. He shows that this problem can be formulated as an assignment problem, therefore it can be solved using the method proposed in Panwalkar and Rajagopalan (1992). For a special case where all jobs have a common difference between normal and crash processing time (a similar assumption to the one in Biskup and Jahnke (2001)) and an equal unit compression penalty, he describes an efficient algorithm that runs in $O(n \log n)$.

Chen and Sheen (2007) extend the TWET problem to a multi-objective approach similar to the one reported in Lee, Danusaputro and Lin (1991), however, they remove the agreeable rates condition and the job-dependent weights for tardy jobs. They propose a branching procedure that finds pareto-optimal solutions for any possible number of tardy jobs. A bi-objective approach is also seen in Giannopoulos and Nearchou (2018), who consider the TWET problem with asymmetric weights and total flow time as minimization criteria. They show a multi-objective DE algorithm devoted to search pareto-optimal solutions, which is an adaptation of the DE reported in Nearchou (2008). A performance comparison is made with other metaheuristics. Table 3 and 4 gives the summary of algorithms for single-machine problems with a TWET objective function.

2.2.4 Common due windows

If a common due date with tolerances u and v is given, symmetric and asymmetric due windows can be distinguished. A common due window is said asymmetric if $u \neq v$ and symmetric otherwise. Occasionally, a situation similar to the symmetric case exists if there is no common due date with tolerances, instead, a time interval (d_l, d_r) designates the period within which no penalties incur. If the penalties are calculated from the boundaries, the penalty function is said to be continuous. On the other hand, it is said to be discontinuous if the penalties are calculated from the common due date. Moreover, a common due window is called small if only one job can avoid penalties. If this technical restriction is not required, the common due window is called large.

Cheng (1988b) addresses the case of small symmetric tolerance intervals by imposing $2u < \min_{\forall j \in N}\{p_j\}$. He shows that the optimal solution can be found in $O(n \log n)$ for the MAD problem. Later, Dickman, Wilamowsky and Epstein (1991) conclude that there are alternative optima that Cheng (1988b) neglected. Wilamowsky, Epstein and Dickman (1996) also consider the discontinuous penalty function in Cheng (1988b), however, extends it to tolerance intervals of any size ($2u \geq \min\{p_j\}$). They describe efficient procedures for the MAD as well as WSAD problems.

For cases of continuous penalty functions, Kramer and Lee (1993) consider a large symmetric common due window for the WSAD problem. They show that the problem is polynomially solvable if the due window is a decision variable. In case of the window as given parameter, they show a dynamic programming algorithm that runs in $O(d_l n^2 + (d_r - d_l)n)$, which then was improved by Weng and Ventura (1996a). Weng and Ventura (1994) proposes an efficient constructive algorithm for the MAD problem with a large due window. In a follow-up paper, Ventura and Weng (1996) deal with the restricted version of the problem and propose a Lagrangean relaxation procedure and two heuristics. Other variation of the MAD problem is given in Weng and Ventura (1996b), where each job may have a different symmetric tolerance ($u = v$) around the due date. They show a pseudo-polynomial dynamic programming algorithm, which can also be used to find a lower bound, as well as heuristic.

Liman, Panwalkar and Thongmee (1996) deals with the problem introduced by Kramer and Lee (1993) and proves that it remains polynomially solvable if the due window is a decision variable that incurs location penalties. Liman, Panwalkar and Thongmee (1997) consider the WSAD problem when the common due window needs to have its size determined. Furthermore, they also incorporate controllable processing times to the objective function and prove that it can be formulated as an assignment problem. Hence, it can be solved in polynomial time. Later, Liman, Panwalkar and Thongmee (1998) generalize the problem to the case when both the size and location of the common due window are decision variables and show an efficient procedure that runs in $O(n \log n)$ to find optimal solutions.

For the unrestricted TWET problem with a large due window, Azizoglu and Webster (1997a) show a branch-and-bound algorithm by which they are only able to solve instances with up to 20 jobs to optimality. The restricted version, was then addressed by Biskup and Feldmann (2005) who provide a complete mapping of the problem, which takes advantage of inherent structural properties. They tackle the problem by means of metaheuristic and also show a constructive heuristic to find initial solutions. It is worth mentioning that for the case with large restrictive due window for the unrestricted TWET with a large common due window, property 1 hold, and property 2 and 3 take the form, respectively (BISKUP; FELDMANN, 2005):

- **Property 9**: An optimal schedule is V-shaped. Jobs with $C_j \leq d_l$ are sequenced in non-increasing order of the ratio p_j/α_j and jobs starting no earlier than d_r are sequenced in nondecreasing order of the ratio p_j/β_j.

- **Property 10**: There exists an optimal schedule in which one job completes precisely at d_l or d_r.

For the restricted case, property 1 and 9 hold and property 5 takes the form (BISKUP; FELDMANN, 2005):

❑ **Property 11**: In an optimal schedule, there exists a job that either starts at time zero or completes precisely at d_l or d_r.

Ying, Lin and Lu (2017) propose an efficient dispatching rule and a constructive heuristic that runs in $O(n^2)$. They test the efficiency of their procedure by incorporating the solutions of the heuristic with the metaheuristics in Biskup and Feldmann (2005). According to the benchmarks, they found better solutions in 144 of 250 instances. More recently, Lin et al. (2019) map 12 possible solutions for the same problem and propose a backtracking simulated annealing that presents several improvements in relation to other *ad hoc* metaheuristics, specially for large-sized instances ($n = 500$ and $n = 1000$).

Gerstl and Mosheiov (2013a) study a series of single-machine assignment problems where both the start time of the window as well as its size are decision variables with the special consideration of unit-time jobs. Several combinations of decision variables and cost factors are shown to be solved in no more than $O(n^3)$ time.

A more general objective function is seen in Yeung, Oguz and Cheng (2001). They assume that the agreeable rate condition applies ($p_j/\alpha_j < p_k/\alpha_k \Rightarrow p_j/\beta_j \leq p_k/\beta_k$) and consider an objective function to minimize the sum of weighted earliness and tardiness, weighted number of early and tardy jobs as well as a due window location penalty. Several pseudo-polynomial dynamic programming algorithms are shown and polynomially solvable cases are discussed. Table 5 gives the summary of algorithms for single-machine problems with common due windows.

2.2.5 Batching and job families

In many production environments, jobs are delivered to costumers in batches. Herrmann and Lee (1993) address the TWET problem when d is a given restricted parameter. They assume that there is a fixed cost for tardy deliveries. Hence, it may be cheaper to delay shipping a job until the delivery time of the following job, since the delay saves the delivery charge, however, it incurs extra ET costs. All early jobs are delivered in one batch at d. Each tardy batch incurs a fixed cost ϕ regardless of its size. If D_j is the delivery date of job j, then, its earliness and tardiness are defined as $E_j = D_j - C_j$ and $T_j = D_j - d$. It is worth noting that a tardy job can incur an earliness penalty (holding cost) if it is not delivered precisely after its completion. They propose dynamic programming algorithms, some heuristics, and discuss some polynomially solvable cases. Later, Yuan (1996) proves that the general problem introduced by Herrmann and Lee (1993) is strongly NP-hard. Chen (1996) extends it to the context when d is a decision variable and shows an algorithm that runs in $O(n^5)$ time that finds an optimal solution.

Azizoglu and Webster (1997b) deal with the unrestricted TWET problem with job families. A set-up time s_i occurs to process a job from family i after a job from other family. No set-up is required to process jobs from the same family. A branch-and-bound

Table 5 – Summary of algorithms for single-machine problems with common due windows

Problem	Complexity	Reference	Algorithm
$1\|\langle d(\pm u)\rangle\|\sum(E_j+T_j)$	P	Cheng (1988b)	$O(n\log n)$
$1\|\langle d(\pm u)\rangle\|\sum(E_j+T_j)$	P	Wilamowsky, Epstein and Dickman (1996)	$O(n\log n)$
$1\|\langle d-u,d+v\rangle\|\sum(E_j+T_j)$	P	Wilamowsky, Epstein and Dickman (1996)	$O(n\log n)$
$1\|\langle d(\pm u)\rangle\|\sum(\alpha E_j+\beta T_j)$	P	Wilamowsky, Epstein and Dickman (1996)	$O(n\log n)$
$1\|\langle d_l,d_r\rangle\|\sum(E_j+T_j)$	P	Weng and Ventura (1994)	$O(n\log n)$
$1\|\langle d_l,d_r\rangle\|\sum(\alpha E_j+\beta T_j)$	P	Kramer and Lee (1993)	$O(n\log n)$
$1\|\langle d_l,d_r\rangle\|\sum(\alpha E_j+\beta T_j+\gamma d_l)$	P	Liman, Panwalkar and Thongmee (1996)	$O(n\log n)$
$1\|\langle d_l,d_r\rangle\|\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta d_r)$	P	Liman, Panwalkar and Thongmee (1998)	$O(n\log n)$
$1\|\langle d_l,d_r\rangle\|\sum(\alpha E_j+\beta T_j+\lambda_j(x_j)+\gamma d_l+\delta d_r)$	P	Liman, Panwalkar and Thongmee (1997)	$O(n^3+n^2\log n)$
$1\|\langle d(\pm u_j)\rangle\|\sum(E_j+T_j)$	NP-hard	Weng and Ventura (1996b)	*DP*
			Heuristic
$1\|\langle \hat{d}_l,\hat{d}_r\rangle\|\sum(E_j+T_j)$	NP-hard	Weng and Ventura (1994)	
			Heuristic
$1\|\langle \hat{d}_l,\hat{d}_r\rangle\|\sum(\alpha E_j+\beta T_j)$	NP-hard	Kramer and Lee (1993)	*DP*
		Weng and Ventura (1996a)	*DP*
$1\|\langle d_l,d_r\rangle\|\sum(\alpha_j E_j+\beta_j T_j)$	NP-hard	Azizoglu and Webster (1997a)	*Enumer.*
$1\|\langle \hat{d}_l,\hat{d}_r\rangle\|\sum(\alpha_j E_j+\beta_j T_j)$	NP-hard	Biskup and Feldmann (2005)	*Heuristic*
		Ying, Lin and Lu (2017)	*Heuristic*
		Lin et al. (2019)	*Heuristic*

algorithm is proposed for solving instances with up to 15 jobs in size. For larger instances, a heuristic procedure is also described. In their heuristic procedure, at each level of the branch tree, only a limited number of nodes are explored. By choosing the number of nodes, also referred to as beam width, the user can evaluate the trade-off between solution quality and computational cost. Later, Webster, Jog and Gupta (1998) propose a genetic algorithm for solving larger instances of the same problem.

Chen (1997) also considers the TWET problem with job families, however, he imposes that each job family has its own common due date, which can be externally given or internally determined. Set-up times are required for processing jobs from different families and are sequence dependent with the assumption that they satisfy the triangle inequality (MONMA; POTTS, 1989). He shows that the problem with common due dates as given parameters is NP-hard even for the unrestricted version and two batches. He shows dynamic programming algorithms for both cases when the common due date is the same for all job families.

Again, when the common due date is large enough to do not constrain the problem, Suriyaarachchi and Wirth (2004) approach the problem of scheduling job families with sequence independent set-up tasks between them. Their formulation encompass the WSAD objective function for which they propose seven structural properties that are necessary but not sufficient conditions for a schedule to be optimal. Moreover, they show a heuristic procedure inspired by the total flow time minimization problem, a lower bound, a greedy heuristic, and a genetic algorithm.

Yin et al. (2012) report a more general case, which addresses the scheduling and delivery of jobs as well as determining an optimal due date. In addition, they also consider the option of performing a rate-modifying activity on the machine. The processing time of a job scheduled after the rate-modifying activity decreases based on a job dependent factor, and the finished jobs are delivered in batches. However, they do not impose a capacity limit on the delivery batches and the cost per batch delivery is fixed independent of the number of jobs in the batches. To solve the problem, they show some properties of the optimal schedule and present polynomial time algorithms for some special cases. Note that, in contrast with Herrmann and Lee (1993) who consider the holding and earliness cost the same, here, the holding cost is calculated as $H_j = D_j - C_j$ and the earliness cost as $E_j = \max\{0, d - C_j\}$.

Li et al. (2015) report the case of a single batch processing machine that has a fixed capacity that cannot be exceeded. They show several optimal properties for the MAD problem to develop a heuristic procedure to form batches. The heuristic is then incorporated into a genetic algorithm, which is compared with other metaheuristics. Results show that their method outperforms other benchmark methods used for comparison.

Parsa, Karimi and Husseini (2017) also consider a similar problem to the one reported in Li et al. (2015). To find an optimal schedule for a predetermined set of batches,

they show a dynamic programming algorithm. Moreover, they show heuristics, a lower bounding procedure, and a branch-and-bound algorithm. They also argue that property 1 hold and properties 2 and 3 take the form:

- **Property 12**: There exists an optimal schedule in which one batch completes precisely at d.
- **Property 13**: For a given set of formed batches, it is optimal to sequence non-tardy batches in a batch weighted longest processing time (BWLPT) rule and tardy batches in a batch weighted shortest processing time (BWSPT) order such as $\frac{P_1}{n_1} \geq \frac{P_2}{n_2} \geq ... \geq \frac{P_e}{n_e}$ and $\frac{P_{e+1}}{n_{e+1}} \leq \frac{P_{e+2}}{n_{e+2}} \leq ... \leq \frac{P_{e+t}}{n_{e+t}}$, here, $P_{b'}$ and $n_{b'}$ represents the processing time and the number of jobs processed in batch b', respectively, and e and t the number of nontardy and tardy batches, respectively.

Zhao and Li (2008) address the WSAD problem with unbounded batching subject to a common due window. Again, they show several properties that constitutes an optimal schedule. They consider the cases of a given due window location as well as the case of optimizing a its location. Efficient algorithms are also shown for both cases with running times of $O(n^3)$ and $O(n^4)$, respectively. The complexity of the two algorithms can be further reduced to $O(n^2 \log n)$ and $O(n^3 \log n)$, respectively, using geometric techniques (HOESEL; WAGELMANS; MOERMAN, 1994).

Table 6 – Summary of polynomially solvable cases of single-machine problems with batching

Problem	Complexity	Reference	Algorithm
$1\|\langle d_l, d_r\rangle, \mathbf{B}\|\sum(\alpha E_j + \beta T_j)$	P	Zhao and Li (2008)	$O(n^3)$
$1\|\langle d_l, d_r\rangle, \mathbf{B}\|\sum(\alpha E_j + \beta T_j) + \gamma d_l$	P	Zhao and Li (2008)	$O(n^4)$
$1\|\langle d_l, d_r\rangle, \mathbf{B}\|\sum(\alpha E_j + \beta T_j + \theta' H_j + \gamma d_l + \delta d_r) + \phi l$	P	Yin et al. (2013)	$O(n^8)$
			$O(n^5)$
			$O(n^5)$
			$O(n^6)$
			$O(n^4)$
			$O(n^2)$
$1\|\langle d\rangle, \mathbf{B}\|\sum(\alpha E_j + \beta T_j) + \phi l + \gamma d$	P	Chen (1996)	$O(n^5)$

Yin et al. (2013) reports a batch delivery problem in which jobs have an assignable common due window. The objective consists of determining the optimal size and location of the due window, the optimal dispatch date for each job, as well as an optimal job sequence that minimizes a cost function composed of earliness, tardiness, holding time, window location and size, and batch delivery. The authors show the characteristics of optimal schedules and develop a dynamic programming algorithm that runs in $O(n^8)$.

Mönch, Unbehaun and Choung (2006) address the scheduling problem for a single burn-in oven in the semiconductor manufacturing industry, where the oven is a batch processing machine with a restricted capacity. The processing time of a batch should be set as the longest processing time among all of the jobs contained in the batch. The objective is to minimize the MAD under the constraint that the maximum tardiness should be less than or equal to the maximum allowable time value. To tackle this problem, the authors show heuristics based on dominance properties, which construct a solution without considering the maximum allowable tardiness constraint. If this solution does not satisfy the stipulated tardiness bound, a second phase is applied to satisfy the condition. Tables 6 and 7 give the summary of algorithms for single-machine problems with batching.

Table 7 – Summary of NP-hard cases of single-machine problems with batching

Problem	Complexity	Reference	Algorithm
$1\|\langle\hat{d}\rangle, \mathbf{B}\|\sum(\alpha_j E_j + \beta_j T_j) + \phi l$	NP-hard	Herrmann and Lee (1993)	*DP*
			$O(n^2)$
$1\|\langle d_j\rangle, f, ST_{sd,b}\|\sum(\alpha_j E_j + \beta_j T_j) + \gamma d_j$	NP-hard	Chen (1997)	$O(n^8)$
			$O(n^4)$
$1\|\langle d\rangle, f, ST_{sd,b}\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Chen (1997)	$O(n^4)$
$1\|\langle d\rangle, f, ST_{si,b}\|\sum(\alpha E_j + \beta T_j)$	NP-hard	Suriyaarachchi and Wirth (2004)	*Heuristic*
$1\|\langle d\rangle, f, ST_{si,b}\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Azizoglu and Webster (1997b)	*Enumer.*
			Heuristic
$1\|\langle d\rangle, \mathbf{B}\|\sum(E_j + T_j)$	NP-hard	Li et al. (2015)	*Enumer.*
			Heuristic
		Parsa, Karimi and Husseini (2017)	*Enumer.*
			DP
			Heuristic
$1\|\langle d\rangle, \mathbf{B}, rm\|\sum(\alpha E_j + \beta T_j + \theta' H_j + \gamma d) + \phi l$	Open	Yin et al. (2012)	$O(n^6)$
			$O(n^4)$
			$O(n^5)$
			$O(n^5)$
$1\|\langle d\rangle, \mathbf{B}\|\sum(E_j + T_j)$ s.t. $\bar{T}_{\max}$	Open	Mönch, Unbehaun and Choung (2006)	*Heuristic*

2.2.6 Learning, ageing, and deterioration effects

Learning effects might be an important factor to consider in short term production planning. Scheduling decisions may also take into consideration these effects due to the inherent characteristic of high level of human activities in most production environments. Biskup (1999) was one of the pioneers in dealing with ET scheduling with learning effects and a common due date. He addresses the WSAD problem with additional completion time penalties. In this approach, it is assumed that a normal processing time p_j of job j is incurred if the job is scheduled first in the sequence. However, the processing times tend to decrease because of the learning effect, that is, the processing time of job j if it is scheduled in a certain position changes. Here, let it be denoted as position r'. Then, the processing time for job j in position r' of a sequence is $p_{jr'} = p_j r'^L$, where $L \leq 0$, is the learning index given as the logarithm to the base 2 of the learning rate. Biskup (1999) shows that the problem can be solved in $O(n^3)$ time in two steps. First, an assignment problem which leads to an optimal sequence of the jobs solved. Second, the optimal due date is determined like in the standard problem without learning considerations since the introduction of these effects does not seem to have an influence on the number of early and tardy jobs. Mosheiov and Sidney (2003) deal with the problem reported by Panwalkar, Smith and Seidmann (1982) and show that it can also be solved in $O(n^3)$ if a job-dependent learning effect is considered, that is, each job may have its own learning index L_j.

Biskup and Simons (2004) assume that the learning effect can be controlled. If no learning effort takes place, a standard learning rate occurs. However, it is also possible to speed up learning in such a way that a greater reduction on the processing times of later jobs might be achievable. Now let, s denote the standard learning rate. A decrease in the learning rate u' is possible with an increase in learning costs. If x is the percentage reduction of s, thus $u' = (1 - x)s$. $k(x)$ are the costs which are incurred by the learning efforts to achieve x, which is assumed to be a convex function. With that said, the processing time of job j in position r' is given as $p_{jr'} = p_j r'^{\log_2 u'}$, with the understanding that $0 \leq x \leq x_{\max} < 1$. Therefore, the objective function is to minimize $\sum_{j=1}^{n} (E_j + T_j) + k(x)$, which they show to be polynomially solvable in $O(n^3)$.

Wang and Wang (2010) outline the case of job-dependent learning effects and multiple common due dates. In their approach, the job processing time is given as $p_{jr'} = p_j r'^{L_j}$ and groups of jobs should be scheduled to different common due dates (d_j). The objective is to minimize the costs incurred with earliness, tardiness as well due-date assignment penalties. They show that, when the number of jobs assigned to each due date is externally specified, the problem can be solved using the well-know Hungarian method in $O(n^3)$ time.

Wang and Wang (2011) address the case with both learning and deterioration effects. This characteristic implies that the processing time is given as $p_{jr'} = (p_j + (\omega \times S_j))r'^L$, where S_j is the starting time of job j and ω is a constant representing the deterioration

effect. Furthermore, they consider the assignment of a common due window rather than a common due date. This problem also remains polynomially solvable with solutions found in $O(n \log n)$.

Wang and Wang (2014) study the due window assignment problem with controllable processing times involving a linear as well as a convex resource consumption function. They also incorporate job-dependent learning effects. They show algorithms that run in $O(n^3)$ for both linear and convex resource allocation functions as well as an $O(n \log n)$ procedure for the special case when the learning index is common to all jobs.

Cheng, Kang and Ng (2004) studied the assignment of a common due date with a linear deterioration effect and concluded that the optimal solution may be found in $O(n \log n)$. Li, Ng and Yuan (2011) also consider a linear deterioration effect, however, they report an unusual objective function that assesses penalties for earliness, due date assignment as well as the weighted number of tardy jobs. The more general case is solvable in $O(n^4)$, while the special case that imposes restrictions for the deterioration rate is solvable in $O(n^3)$. A similar problem with due window assignment is discussed by Liu, Wang and Min (2014) who propose a $O(n^2 \log n)$ algorithm when there are penalties for earliness, tardiness as well as for the start of the due window (d_l).

Other worth noting class of scheduling problems with processing time considerations is the addition of a rate-modifying activity. This activity usually affects the performance of a machine, i.e., it affects the production time of the following jobs. Quite often, the rate modification is due to maintenance, suggesting that the machine will improve together with its production rate. Mosheiov and Oron (2006) was one of the first to consider a due date assignment with ET penalties and maintenance. They show that the problem can be solved in $O(n^4)$ time. For the common due window assignment problem, a solution procedure of the same complexity order is provided in Mosheiov and Sarig (2009).

An extension of the problem reported in Mosheiov and Sarig (2009) is seen in Yang (2010) who simultaneously include star-time dependent learning and position dependent ageing effect under deteriorating maintenance. Now, assume that $p_{jr'}$ is the actual processing time of job j scheduled in position r', then, $p_{jr'} = p_j - \omega S_j)r'^L$, where S_j is the start time of job j, L is an ageing factor ($L \geq 0$), and $\omega > 0$ is a common decreasing rate. An $O(n^2 \log n)$ exact algorithm was provided. In other study, Yang, Yang and Cheng (2010) consider a slightly different processing time model, where the processing time of job j is given as $p_{jr'} = p_j r'^{L_j}$, where $L_j > 0$ is a job-dependent ageing factor. Assuming this configuration, an exact solution is possible in $O(n^4)$. If $L_j = L$, a lower-order exact algorithm that runs in $O(n^2 \log n)$ is also available.

Cheng, Yang and Yang (2012) assume that the duration of the maintenance activity is not constant. Their due window assignment model supposes that the job processing time is given by a linear time-dependent deteriorating function such that the actual processing time is $p_j = \hat{p}_j + \omega S_j$, where $\omega > 0$ is the deterioration rate and $\hat{p}_j$ is the normal processing

time of job j. Here, the rate modifying activity is allowed to improve efficiency, but rather than inducing a modifying rate Ψ_j, it is assumed that the machine would be restored to its initial condition and the deterioration effect would start anew. The maintenance durations is also given by a linear function and it is allowed only once throughout the scheduling horizon. An optimal solution for this problem can be found in $O(n^2 \log n)$. A very similar approach is seen in Zhao and Tang (2012), however, instead of restoring the initial condition of the machine, the rate modifying activity changes the actual processing time such that $p_j = \Psi_j \hat{p}_j + \omega S_j$. The authors provide an $O(n^4)$ time solution for the problem.

Zhu et al. (2011) studies the due window assignment problem with multiple rate-modifying activities as well as learning and deterioration effects. The actual processing time is given as $p_{jr'} = p_j r'^{L_j} + \omega S_j$ where $L_j \leq 0$ and $\omega > 0$ are the learning index and deterioration rate, respectively. The multiple rate-modifying activities are assumed to have a constant duration. If job j is sequenced in position r' after this rate-modifying activity y, its actual processing time is now given as $p_{jr'} = \Psi_{jy} p_j r'^{L_j} + \omega S_j$. The proposed solution procedure runs in $O(n^{3+\bar{u}})$, where $\bar{u}$ is the number of rate-modifying activities in the scheduling horizon.

Fan and Zhao (2014) describes a problem with multiple common due dates, ageing effects and a deteriorating maintenance activity. The number of common due date is a given parameter and the objective is to find the position as well as the set of jobs assigned to each due date. The actual processing time of job j scheduled in position r' is given as $p_{jr'} = p_j r'^{L_j}$, where $L_j > 0$. The maintenance duration increases linearly according to its start time. Moreover, the authors assume that after a maintenance operation is performed, the ageing effect will start anew. For this problem, they show an algorithm that runs in $O(n^4)$.

Mor and Mosheiov (2015) study several different maintenance possibilities and common due window assignment. When the length of the maintenance is a linear function of its starting time, i.e., $t_0 > 0$ is the basic starting time of a maintenance operation, then its length is given as $T_{MA} = t_0 + \omega t$ where $\omega > 0$ is a non-negative deterioration factor. They also consider a learning effect such that the length is now given as $T_{MA} = t_0 - \omega t$. Moreover, they assume that the maintenance can be a function of its position: if it is scheduled before the job in position r', it requires $T_{MA}(r') = \tau_{r'}$ time, where $\tau_{r'}$ is a position-dependent constant, and $\tau_{r'} \leq \tau_{r'+1}$, $r' = 1, ..., n-1$. Both position-dependent (MOSHEIOV; SARIG, 2008) and position-independent processing times are investigated and an exact solution procedure is available in $O(n^4)$ time.

Ji et al. (2013) propose a general model of due date assignment that encompasses resource allocation, ageing, and a deteriorating rate-modifying activity. Both linear and convex resource consumption functions are described. The rate-modifying activity duration increases linearly according to its starting time and there is also the assumption

Table 8 – Summary of algorithms for single-machine problems with learning, ageing, and deterioration

Problem	Complexity	Reference	Algorithm
$1\|\langle d\rangle, lrn\|\sum(\alpha E_j+\beta T_j+\theta C_j)$	P	Biskup (1999)	$O(n^3)$
$1\|\langle d\rangle, lrn\|\sum(\alpha E_j+\beta T_j+\gamma d)$	P	Mosheiov and Sidney (2003)	$O(n^3)$
$1\|\langle d\rangle, lrn\|\sum(E_j+T_j)+k(x)$	P	Biskup and Simons (2004)	$O(n^3)$
$1\|\langle d_j\rangle, lrn\|\sum(\alpha E_j+\beta T_j+\gamma d)$	P	Wang and Wang (2010)	$O(n^3)$
$1\|\langle d\rangle, dtr\|\sum(\alpha E_j+\beta T_j+\gamma d)$	P	Cheng, Kang and Ng (2004)	$O(n\log n)$
$1\|\langle d\rangle, dtr\|\sum(\alpha E_j+\alpha' U_j+\gamma d)$	P	Li, Ng and Yuan (2011)	$O(n^4)$
			$O(n^3)$
$1\|\langle d_l, d_r\rangle, dtr\|\sum(\alpha E_j+\beta T_j+\gamma d_l)$	P	Liu, Wang and Min (2014)	$O(n^2\log n)$
$1\|\langle d\rangle, lrn, dtr\|\sum(\alpha E_j+\beta T_j+\gamma d)$	P	Kuo and Yang (2011)	$O(n^3)$
			$O(n\log n)$
$1\|\langle d_l, d_r\rangle, lrn, dtr\|\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta d_r)$	P	Wang and Wang (2011)	$O(n\log n)$
$1\|\langle d_l, d_r\rangle, lrn\|\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta(d_r-d_l))+\rho\lambda_j(x_j)$	P	Wang and Wang (2014)	$O(n^3)$
			$O(n^3)$
			$O(n\log n)$

Table 9 – Summary of algorithms for single-machine problems with rate modifying activities

Problem	Complexity	Reference	Algorithm
$1\|\langle d\rangle, rm\|\sum(\alpha E_j+\beta T_j+\gamma d)$	P	Mosheiov and Oron (2006)	$O(n^4)$
$1\|\langle d_l,d_r\rangle, rm\|\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta(d_r-d_l))$	P	Mosheiov and Sarig (2009)	$O(n^4)$
$1\|\langle d_l,d_r\rangle, dtr, lrn, dtr(rm)\|$ $\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta(d_r-d_l))$	P	Yang (2010)	$O(n^2\log n)$
			$O(n\log n)$
$1\|\langle d_l,d_r\rangle, dtr, dtr(rm)\|$ $\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta(d_r-d_l))$	P	Yang, Yang and Cheng (2010)	$O(n^4)$
			$O(n^2\log n)$
		Cheng, Yang and Yang (2012)	$O(n^2\log n)$
		Zhao and Tang (2012)	$O(n^4)$
$1\|\langle d_l,d_r\rangle, dtr, lrn, mrm\|$ $\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta(d_r-d_l))$	P	Zhu et al. (2011)	$O(n^{3+\bar{u}})$
$1\|\langle d_j\rangle, dtr, dtr(rm)\|\sum(\alpha E_j+\beta T_j+\gamma d_j)$	P	Fan and Zhao (2014)	$O(n^4)$
$1\|\langle d_l,d_r\rangle, dtr(rm)\|\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta(d_r-d_l))$	P	Mor and Mosheiov (2015)	$O(n^4)$
$1\|\langle d_l,d_r\rangle, dtr, dtr(rm)\|$ $\sum(\alpha E_j+\beta T_j+\gamma d_l+\delta(d_r-d_l))+\rho\lambda_j(x_j)$	P	Ji et al. (2013)	$O(n^4)$
		Cheng and Cheng (2014)	$O(n^2\log n)$

that after the rate-modifying activity is performed, the machine will return to its initial condition. An $O(n^4)$ solution procedure is possible for both convex and linear resource consumption models. Note that they also propose an $O(n^2 \log n)$ solution for the case of a convex resource consumption when all the ageing factors are identical, however, Cheng and Cheng (2014) claim that this procedure is only valid when the modifying rate is unitary, i.e., $\Psi_j = 1$, for $j = 1, 2, ..., n$. They also propose some corrections to the special-case algorithm of Ji et al. (2013).

Tables 8 gives the summary of algorithms for single-machine problems with learning, ageing, and deterioration effects and Table 9 gives the summary of algorithms with rate-modifying activities.

2.2.7 Non-linear cost functions

Another way to measure the deviations around a common due date is to consider that large deviations are highly discouraged. In other words, only small deviations of completion times from the due date are acceptable. An appropriate way of establishing a performance measure in this context is to take into account the mean squared deviations (MSD) objective function:

$$f(\pi, d) = (1/n) \sum_{j=1}^{n} (E_j^2 + T_j^2) \tag{4}$$

Bagchi, Sullivan and Chang (1987) show the equivalence of the unrestricted version of the MSD problem with the completion time variance problem (CTV) (MERTEN; MULLER, 1972). They show a solution procedure that generates multiple optimal solutions, however, determining the minimal one may require additional efforts. In case of an unrestricted d, properties 1, 2, and 3 hold. The following property also holds (BAGCHI; SULLIVAN; CHANG, 1987):

- ❑ **Property 14**: For any schedule, the optimal due date d that minimizes the MSD is equal to the mean completion time $\bar{C} = (1/n) \sum_{j=1}^{n} C_j$.

In relation to the restricted version of the problem, Bagchi, Sullivan and Chang (1987) show an enumerative algorithm that imposes all schedules to start at time zero, which restricts their search for optimal solutions. Bagchi, Chang and Sullivan (1987) provide a similar procedure when there are different ET weights. Later, De, Ghosh and Wells (1989) and De, Ghosh and Wells (1990b) remove the unnecessary assumption of a zero start time. Note that the CTV problem is NP-hard, therefore, the MSD problem is also NP-hard (KUBIAK, 1993).

De, Ghosh and Wells (1993) consider the objective function with different, but not job-dependent, ET weights and develop a dynamic programming algorithm for both restricted and unrestricted cases. Kahlbacher (1993) addresses the more general case

$\sum_{j=1}^{n}(\alpha E_j^c + \beta T_j^c)$, where c is a real exponent with $c \geq 1$. A pseudo-polynomial dynamic programming solution is given when d is restricted.

Mondal (2002) also rely on dynamic programming to solve the MSD problem. In contrast with formulations that consider restricted and unrestricted versions separately, Mondal (2002) provide an unified pseudo-polynomial algorithm that outputs optimal solutions regardless of the restrictiveness factor of the due date. A heuristic that produces near-optimal solutions is also described. More recently, Srirangacharyulu and Srinivasan (2013) also propose an exact procedure that runs in $O(nd)$ that works irrespective of the version of the problem.

The case with symmetric job-dependent weights is addressed by Pereira and Vasquez (2017). They show that the unrestricted problem is also equivalent to the weighted CTV and several dominance properties are established. Pereira and Vasquez (2017) rely on a branch-and-cut approach to solve a time-indexed formulation. The branch-and-cut procedure is further enhanced with a primal heuristic, a branching scheme that takes advantage of the several dominance properties, and a variable prefixing procedure. Optimal solutions can be found for instances with up to 300 jobs for due dates with different tightness values. Table 10 gives the summary of algorithms for single-machine problems with a MSD objective function.

Table 10 – Summary of single-machine problems with non-linear cost functions

Problem	Complexity	Reference	Algorithm
$1\|\langle\hat{d}\rangle\|\sum(E_j^2+T_j^2)$	NP-hard	Bagchi, Sullivan and Chang (1987)	*Enumer.*
		De, Ghosh and Wells (1990b)	*Enumer.*
			Approx.
		De, Ghosh and Wells (1993)	*DP*
		Mondal (2002)	*DP*
		Mondal (2002)	*Heuristic*
		Srirangacharyulu and Srinivasan (2013)	*Enumer.*
$1\|\langle\hat{d}\rangle\|\sum(\alpha E_j^2+\beta T_j^2)$	NP-hard	Bagchi, Chang and Sullivan (1987)	*Enumer.*
		De, Ghosh and Wells (1993)	*DP*
		Kahlbacher (1993)	*DP*
$1\|\langle\hat{d}\rangle\|\sum\alpha_j(E_j^2+T_j^2)$	NP-hard	Pereira and Vasquez (2017)	*Enumer.*

2.3 Parallel-machine models

The analysis of the problems involving a single-machine has been extended to models with multiple parallel-machines. Again, there are several well-established properties that constitute optimal solutions for both restricted and unrestricted versions of the problem. This section is organized as follows: first we introduce the basic problem settings (MAD,

WSAD, and TWET), then, we extend the discussion to cases with a common due window. Since there is a very limited number of papers dealing with ET scheduling in parallel machines with batching, we decided to omit them from this chapter. The same occurs with non-linear cost functions. Finally, we consider papers dealing with learning effects or other processing time considerations. Again, Tables 10 - 12 present the summary of algorithms for each subsection.

2.3.1 Mean absolute deviation

The results reported from the basic analysis of the unrestricted single-machine models have been extended to the ones that consider multiple parallel machines when the due date is a decision variable (SUNDARARAGHAVAN; AHMED, 1984; HALL, 1986). The standard procedure to solve the problem consists of assigning the m longest jobs to different machines. Subsequently, the jobs are sorted in non-increasing order of processing times and then distributed $2m$ at a time to each machine. Finally, the algorithm in Bagchi, Sullivan and Chang (1986) can be used to obtain the job sequence in each machine. Note that, for the multiple parallel machines case, this procedure produces all alternative optimal solutions, but the minimum due date cannot be established beforehand. Rather, all alternative optima must be found and their due dates compared. Furthermore, properties 1, 2, 3, and 4 take the following form (BAKER; SCUDDER, 1990):

- ❑ **Property 15**: There is no inserted idle time on each machine.
- ❑ **Property 16**: The optimal schedule is V-shaped on each machine.
- ❑ **Property 17**: One job completes precisely at d on each machine.
- ❑ **Property 18**: The number of jobs assigned to each of the m machines is the integer portion of $\frac{n}{m}$ or $\frac{n}{m}+1$. Let q denote this number. Then, the bth job in sequence completes precisely at d. Where b is the smallest integer greater than or equal to $q/2$.

Alidaee and Ahmadian (1993) are the first to consider the MAD problem on an unrelated parallel machine environment. They show that the problem of finding a due date and a schedule is polynomially solvable by reducing it to a transportation problem. In addition, they also show a heuristic to minimize the makespan of all schedules. Emmons (1987) addresses both cases of parallel and uniform machines and proposes a polynomial time algorithm. Again, he considers some secondary criteria to choose among various schedules such as the minimization of makespan and the accomplishment of a minimal due date. It is necessary to separate the problem of finding the optimal due date from the problem of finding the minimal one among optimal due dates. In contrast with single machine models, these problems differ in complexity.

In a bi-objective approach, Su (2009) addresses the MAD problem in an identical parallel machines context subject to the minimum flow time. It is shown that the problem can be transformed into a simplified version of the problem with objective to minimize makespan subject to minimum flow time. Several properties are established to solve optimally the restricted version of the problem by adopting a binary integer programming model.

Table 11 – Summary of algorithms for the parallel-machine MAD problem

Problem	Complexity	Reference	Algorithm
$Q\|\langle d\rangle\|\sum(E_j+T_j)$	P	Emmons (1987)	$O(n\log n)$
$R\|\langle d\rangle\|\sum(E_j+T_j)$	P	Alidaee and Ahmadian (1993)	$O(n^3)$
			Heuristic
$P\|\langle\hat{d}\rangle\|\sum(E_j+T_j)/\sum C_j$	NP-hard	Su (2009)	*Enumer.*

2.3.2 Weighted sum of absolute deviations

Emmons (1987) also extend the results reported in the previous subsection to the WSAD context, which remain polynomially solvable. However, the problem of finding a minimal due date and the associated optimal schedule is NP-hard even if $m = 2$ and strongly NP-hard for an arbitrary m (DE; GHOSH; WELLS, 1994a).

Cheng (1989) considers a more general objective function with due date penalty. He shows a heuristic procedure claiming that it is capable of finding near-optimal solutions. Later, De, Ghosh and Wells (1991) argue that the procedure in Cheng (1989) neglects the fact that an optimal schedule does not necessarily starts at time zero on all machines and also that he overlooks the critical V-shape property. To correct these flaws, De, Ghosh and Wells (1991) characterize the conditions for an optimal schedule and show a dynamic programming algorithm. Cheng and Chen (1994) show that the problem of finding a due date and a schedule with penalties assessed for earliness, tardiness, and due date is NP-hard with either a total or a maximal penalty function. They also show that the the special case where all jobs share identical processing times is polynomially solvable. Recently, Drobouchevitch and Sidney (2012) presented a polynomial time algorithm that runs in $O(m^2 \log m)$ for the unrelated parallel machines environments with identical jobs.

Diamond and Cheng (2000) slightly alter the heuristic employed by Cheng (1989) and apply it to the problem without the zero start time constraint and show that it is asymptotically optimal as the number of jobs approaches infinity as long as the number of machines m and the weight parameters γ and β are held constant. Xiao and Li (2002) also propose a heuristic procedure for which they provide an absolute performance ratio. Moreover, they show another heuristic with a better worst-case performance for the case when there is no earliness penalty and a fully polynomial approximation scheme. For

the unrelated parallel machine case, Adamopoulos and Pappis (1998) show a heuristic procedure that runs in $O(n^3)$.

Biskup and Cheng (1999a) consider the identical parallel machine environment where the objective is to minimize a cost function consisting of earliness, tardiness, and completion time penalties. They show that this problem is NP-hard and propose a polynomial time algorithm when all jobs share identical processing times. For the general problem, they introduce a heuristic to find near-optimal solutions.

Alidaee and Panwalkar (1993) address the case of compressible processing times in uniform parallel processors. They show that it can be reduced to an assignment problem, thus, optimal solutions can be found in $O(n^3)$ time.

Min and Cheng (2006) shows several genetic algorithms to determine a common due date and a schedule on identical parallel machines. In order to show the robustness of their approach, they use data from a large textile company. Best results are reported for the hybrid genetic algorithm with simulated annealing as well as the one with a iterative heuristic fine-tuning operator. Kim, Kim and Lee (2012) also rely on metaheuristics to solve the same problem. First, they show a two stage heuristic for obtaining an initial solution. Second, the initial solution serves as an input for metaheuristic. In addition, they also discuss some properties that are important to enhance the search procedure.

Table 12 – Summary of algorithms for the parallel-machine WSAD problem

Problem	Complexity	Reference	Algorithm
$P\|\langle d\rangle, p_j = p\|\sum(\alpha E_j + \beta T_j)$	P	Cheng and Chen (1994)	$O(1)$
$Q\|\langle d\rangle\|\sum(\alpha E_j + \beta T_j)$	P	Emmons (1987)	$O(n \log n)$
$R\|\langle d\rangle, p_j = p\|\sum(\alpha E_j + \beta T_j + \gamma d)$	P	Drobouchevitch and Sidney (2012)	
$Q\|\langle d\rangle\|\sum(\alpha E_j + \beta T_j + \lambda_{ij}(x_{ij}))$	P	Alidaee and Panwalkar (1993)	$O(n^3)$
$P\|\langle d^{\min}\rangle\|\sum(\alpha E_j + \beta T_j)$	NP-hard	De, Ghosh and Wells (1994a)	*DP*
$P\|\langle d\rangle\|\sum(\alpha E_j + \beta T_j + \gamma d)$	NP-hard	Cheng (1989)	*Heuristic*
		De, Ghosh and Wells (1994a)	*DP*
		Diamond and Cheng (2000)	*Heuristic*
		Xiao and Li (2002)	*Approx.*
		Min and Cheng (2006)	*Heuristic*
		Kim, Kim and Lee (2012)	*Heuristic*
$R\|\langle d\rangle\|\sum(\alpha E_j + \beta T_j + \gamma d)$	NP-hard	Adamopoulos and Pappis (1998)	
$P\|\langle d\rangle\|\sum(\alpha E_j + \beta T_j + \theta C_j)$	NP-hard	Biskup and Cheng (1999a)	*Heuristic*

2.3.3 Total weighted earliness and tardiness

Chen and Powell (1999) were one of the pioneers to study the TWET problem with a common due date in a multiple parallel machine environment. They assume that d is unrestrictedly large and tackle the problem by first formulating it as an integer program and then reformulate it using Dantzig Wolfe decomposition as a set partitioning problem with side constraints. Based on these formulations they propose a branch-and-bound algorithm which is capable of solving instances with up to 60 jobs in size.

The unrelated parallel machine problem with release dates was reported in Bank and Werner (2001). They suggest different constructive algorithms for an heuristic solution of the problem, which are composed of two-stage procedures. The first phase assigns the jobs to machines, and then, for each machine and the corresponding set of jobs, a single machine problem is solved. The authors also proposed some iterative algorithms, which they use for performance comparison.

For the problem when jobs have symmetric weights that are proportional to the respective processing times ($\alpha_j = wp_j$, when w is a given constant), Sun and Wang (2003) provide the NP-hardness proof for the parallel machine environment. They develop a dynamic programming algorithm to reach an optimal solution and two heuristics to tackle the problem with less computational effort. These heuristics are also analysed in relation to their worst-case error bounds. Recently, T'kindt, Shang and Croce (2019) removed the proportional weights condition and integrated a dynamic programming across subsets and machines and a Sort & Search procedure with worst-case time and space complexities in $O(3^n)$. They argue that it was the first worst-case complexity results for non-regular criteria in schedule.

Mosheiov and Yovel (2006) address the particular case of unit processing times ($p_j = 1, \forall j \in N$) with the objective of minimizing $\sum_{j=1}^{n} (\alpha E_j + \beta T_j + \gamma d)$. They provide two efficient algorithms that run in $O(n^4)$ and $O(n^3)$ if $\gamma = 0$.

Plateau and Rios-Solis (2010) use quadratic programming theory to obtain optimal solutions for the asymmetric TWET with both restricted and unrestricted versions of the unrelated parallel machines problem. Their methodology starts by formulating the scheduling problem as 0-1 quadratic programs under linear constraints. These problems are non-convex, however, before submitting them to a branch-and-bound procedure, they reformulate them in such a way that convexity is assured and a high-quality lower bound is obtained. Their results show good solutions for the unrestricted version of the problem, which seem to perform better in relation to the restricted version. Beyranvand, Peyghami and Ghatee (2012) show the deficiencies of the formulation for the restricted version and point out that their model would never generate an optimal solution. To deal with this issue, some improvements that culminate in better solutions are proposed.

Rios-Solis and Sourd (2008) address the identical parallel machine environment for the TWET problem when the common due date is small enough to constrain it. Since this

problem is NP-hard in the strong sense, they propose a neighbourhood search algorithm based on dynamic programming. They argue that starting this search with a simpler but faster algorithm may induce better results. Note that property 14 holds and properties 15 and 16 take the following forms, respectively:

- **Property 19**: In an optimal schedule, there exists a job that either starts at time zero or completes precisely at the due date on each machine.

- **Property 20**: An optimal schedule is V-shaped on each machine. The jobs completed before the due date are ordered according to the non-decreasing ratios p_j/α_j , and the jobs started after the due date are ordered according to non-increasing ratios p_j/β_j.

Later, Alvarez-Valdes, Tamarit and Villa (2015) show two metaheuristic frameworks: path re-linking and scatter search, which combine solutions obtained through a constructive and local search process. To assess the performance of the algorithms, they use the benchmarks reported in Rios-Solis and Sourd (2008) and claim that improved solutions were found. More recently, Lin et al. (2016) test a fast ruin-and-recreate algorithm and compare with the algorithms reported by Alvarez-Valdes, Tamarit and Villa (2015) using the same test set, however, they did not apply the same experiment conditions (the codes were run in different computers with different processors). In their analysis, several improvements were achieved, in some cases, it outperforms previous algorithms in over 65% of the test instances.

2.3.4 Common due windows

Kramer and Lee (1994) were the first to provide an extension from the single machine environment to multiple parallel machines when the due window is not restricted and the weights are job independent and asymmetric. They show that this problem is NP-hard even for the unit weight case and provide a pseudo-polynomial time dynamic programming algorithm when $m = 2$ as well as a heuristic with absolute error bound. The heuristic is extended for the m machine case. The error tends to decrease if the number of jobs is reasonably large.

Chen and Lee (2002) provide a branch-and-bound algorithm for the case when the due window is restrictive and the weights are job-dependent. Their algorithm is based on the column generation approach in which the problem is first formulated as a set partitioning type formulation and then in each branch-and-bound iteration the linear relaxation of this formulation is solved by the standard column generation procedure. However, their algorithm solves instances with up to 40 jobs. In a simplified form, Chen and Lee (2002) show that property 14 holds, and the following properties are true:

Table 13 – Summary of algorithms for the parallel-machine TWET problem

Problem	Complexity	Reference	Algorithm
$P\|\langle d\rangle, p_j = 1\|\sum(\alpha_j E_j + \beta_j T_j)$	P	Mosheiov and Yovel (2006)	$O(n^3)$
$P\|\langle d\rangle, p_j = 1\|\sum(\alpha_j E_j + \beta_j T_j + \gamma d)$	P	Mosheiov and Yovel (2006)	$O(n^4)$
$P\|\langle d\rangle, prp\|\sum\alpha_j(E_j + T_j)$	NP-hard	Sun and Wang (2003)	*DP*
		Sun and Wang (2003)	*Approx.*
$P\|\langle d\rangle\|\sum\alpha_j(E_j + T_j)$	NP-hard	T'kindt, Shang and Croce (2019)	*Enumer.*
$P\|\langle d\rangle\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Chen and Powell (1999)	*Enumer.*
$P\|\langle \hat{d}\rangle\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Rios-Solis and Sourd (2008)	*DP*
		Alvarez-Valdes, Tamarit and Villa (2015)	*Heuristic*
		Lin et al. (2016)	*Heuristic*
$R\|\langle \hat{d}\rangle\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Plateau and Rios-Solis (2010)	*Enumer.*
		Beyranvand, Peyghami and Ghatee (2012)	*Enumer.*
$R\|\langle d\rangle, r_j\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Bank and Werner (2001)	*Heuristic*

- **Property 21**: An optimal schedule is V-shaped on each machine. Jobs with $C_j \leq d_l$ are sequenced in non-increasing order of the ratio p_j/α_j and jobs starting no earlier than d_r are sequenced in non-decreasing order of the ratio p_j/β_j.

- **Property 22**: In an optimal schedule, there exists a job that either starts at time zero or completes precisely at d_l or d_r on each machine.

Mosheiov and Oron (2004) consider the due window assignment problem with unit processing times in identical parallel machines. They show that this problem is solvable in $O(1)$ time and provide several properties of an optimal solution. In a similar problem, but with job-dependent ET weights and an additional penalty for due-window size, (MOSHEIOV; SARIG, 2011) give an efficient solution procedure that runs in $O(n^4)$. Janiak et al. (2012) extended this problem by adding a penalty for the left boundary of the due window and established conditions for optimal schedules. Other special cases that require less computational efforts are described. Gerstl and Mosheiov (2013c) shows that the general problem discussed by Janiak et al. (2012) can be solved by a lower order algorithm in $O(n^3)$ with some improvements.

The case of uniform parallel machines with identical processing times is reported by Mosheiov and Sarig (2010) where they show that the two-machine environment may be solved in constant time. Gerstl and Mosheiov (2013b) assume that the due window can be either restrictive or non-restrictive. It was also assumed that the ET weights are job dependent and the jobs share an identical unit processing time. With that said,

Table 14 – Summary of algorithms for parallel-machine problems with common due windows

Problem	Complexity	Reference	Algorithm
$P\|\langle d_l, d_r\rangle, p_j = 1\|\sum(\alpha E_j + \beta T_j + \gamma d_l + \delta d_r)$	P	Mosheiov and Oron (2004)	$O(1)$
$P\|\langle d_l, d_r\rangle, p_j = 1\|\sum(\alpha_j E_j + \beta_j T_j + \delta(d_r - d_l))$	P	Mosheiov and Sarig (2011)	$O(n^4)$
$P\|\langle d_l, d_r\rangle, p_j = 1\|\sum(\alpha_j E_j + \beta_j T_j + \gamma d_l + \delta(d_r - d_l))$	P	Janiak et al. (2012)	$O(n^5/m^2)$
			$O(n \log n)$
			$O(n^2 \log \frac{n}{m})$
			$O(n^3/m^2)$
		Gerstl and Mosheiov (2013c)	$O(n^3)$
$Q2\|\langle d_l, d_r\rangle, p_{ij} = 1\|\sum(\alpha E_j + \beta T_j + \gamma d_l + \delta(d_r - d_l))$	P	Mosheiov and Sarig (2010)	
$Q\|\langle \hat{d}_l, \hat{d}_r\rangle, p_{ij} = 1\|\sum(\alpha_j E_j + \beta_j T_j)$	P	Gerstl and Mosheiov (2013b)	$O(n^3)$
$P\|\langle d_l, d_r\rangle\|\sum(f_E(E_j) + f_T(T_j)) + f_L(d_l) + f_S(d_r - d_l)$	NP-hard	Janiak et al. (2013)	*DP*
$P2\|\langle d_l, d_r\rangle\|\sum(\alpha E_j + \beta T_j)$	NP-hard	Kramer and Lee (1994)	*DP*
			Approx.
$P\|\langle d_l, d_r\rangle\|\sum(\alpha E_j + \beta T_j)$	NP-hard	Kramer and Lee (1994)	*Approx.*
$P\|\langle \hat{d}_l, \hat{d}_r\rangle\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Chen and Lee (2002)	*Enumer.*

they propose $O(3^m n^3)$ and $O(2^m n^3)$ algorithms for restrictive and non-restrictive settings, respectively.

Janiak et al. (2013) address the problem of due window assignment and scheduling when jobs have independent earliness and tardiness costs, and penalties are also incurred for due window size and location. They establish a number of properties of optimal solutions and derive dynamic programming algorithms, which are pseudo-polynomial if the number of machines is considered to be constant.

2.3.5 Learning, ageing, and deterioration effects

Scheduling literature involving learning effects and other processing time considerations in parallel machine environments is scarce in terms of objective functions that consider ET penalties and common due dates (windows). Cheng, Kang and Ng (2007) address the problem of assigning a common due date to parallel machines with deterioration effects modelled as a linear increasing function of job start times, which is an extension of the single machine case reported in Cheng, Kang and Ng (2004) (see subsection 2.2.6). They show that this problem is NP-hard even for $m = 2$ and propose a heuristic for the general case when penalties are assessed for the common due date. For the special case when $\gamma = 0$, the problem is polynomially solvable.

Toksari and Guner (2008) address the problem with simultaneous effects of learning and deterioration as well as the inclusion of set-up times. The deterioration assumption is similar to the one adopted by Mosheiov (1991). The effects of learning and deterioration mean that the processing time of a job is defined by an increasing function of its starting time and a decreasing function of the position in sequence (see subsection 2.2.6). To tackle the problem, they present a mixed non-linear integer programming approach, however, optimal solutions could be found for instances with only 11 jobs and 2 machines.

Toksari and Guner (2009) consider the possibility to include both linear and non-linear deterioration effects (ALIDAEE; WOMER, 1999). The processing time of job j in position r is given as $p_{jr'} = (p_j + (\omega \times S_j^z))r'^L$, where z stands for the non-linear deterioration effect. To solve the problem, they propose a mathematical model, a heuristic, and a lower bound procedure for cases when d is restricted and unrestricted. In addition, they also conclude that the V-shaped property is valid for an optimal schedule for the WSAD problem. Later, Toksari and Guner (2010a) extend these results to the case o time-dependent learning (KUO; YANG, 2006). The processing time of a job scheduled in position r' is now given as $p_{[r']} = p_{r'} + (\omega \times C^z_{[r'-1]})(1 + \sum_{k=1}^{r'-1} p_{[k]})^L$ with the understanding that $z = 1$ for a linear deterioration. Again, they propose a mathematical model and a heuristic and show that an optimal schedule is also V-shaped.

In another extension of the previous studies, Toksari and Guner (2010b) argued that an optimal schedule is also V-shaped for the TWET problem with learning effects, lin-

ear deterioration, and sequence-dependent set-ups. Again, they show a mathematical formulation for the problem, which is suitable for solving small instances.

Table 15 – Summary of algorithms for parallel-machine problems with learning, ageing and deterioration

Problem	Complexity	Reference	Algorithm
$P\|\langle d\rangle, dtr\|\sum(\alpha E_j + \beta T_j + \gamma d)$	NP-hard	Cheng, Kang and Ng (2007)	*Heuristic*
			$O(n^{m+1}\log n)$
$P\|\langle\hat{d}\rangle, lrn, dtr, ST_{sd}\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Toksari and Guner (2008), Toksari and Guner (2010b)	*Enumer.*
$P\|\langle\hat{d}\rangle, lrn, dtr\|\sum(\alpha E_j + \beta T_j)$	NP-hard	Toksari and Guner (2009)	*Enumer.*
			Heuristic
		Toksari and Guner (2010a)	*Enumer.*
			Heuristic

2.4 The flow-shop environment

In relation to single and parallel-machine environments, there is a relatively small number of papers dealing with ET scheduling and common due dates (windows) in flow shops. The article of Sarper (1995) was certainly one of the pioneers to investigate such a class of problems. Sarper (1995) proposed a mixed integer linear programming model for a two-machine non-permutation flow shop. Since the model was unable to find optimal solutions even for small instances (he reports that a six-job example took more than 75 minutes) some constructive heuristics (H1, H2, and H3) and a scheduling algorithm (used to evaluate the sequences generated by the constructive heuristics) were provided.

Sakuraba, Ronconi and Sourd (2009) also show a mathematical formulation and several heuristics for the problem studied in Sarper (1995), however, they only consider permutation schedules. The heuristics consist of constructing an initial sequence according to several dispatching rules and combining it with a timing algorithm (provides an optimal solution for a given sequence), an insertion procedure as well as a reduced local search. To facilitate the search for good solutions, they derive the following properties for the MAD objective function:

- **Property 23**: A group of consecutive jobs on the second machine can always be delayed generating a gain if it contains more early jobs than non-early jobs and if this delay does not interfere with the rest of the schedule.

- **Property 24**: In an optimal schedule, there is no idle time between jobs that start their processing on the second machine before the due date.
- **Property 25**: In an optimal schedule, a tardy jobs is scheduled to start its processing on the second machine at the maximum time between its completion time on the first machine and the completion time of the previous job on the second machine.
- **Property 26**: No optimal schedule can have more than $n/2$ early jobs.

Sung and Min (2001) also report the two-machine case, however, they allow at least one of the processors to be a batching machine. Three cases are possible: batch-to-discrete, batch-to-batch, and discrete-to-batch sequences. It is worth mentioning that the common due date is not set earlier than the total job processing time in the first machine. The processing time of each batch is assumed to be constant and up to $\bar{c}$ jobs can be grouped and processed simultaneously in each batch. Solution procedures are provided in polynomial time for batch-to-discrete and batch-to-batch, while a pseudo-polynomial dynamic programming procedure is given for discrete-to-batch.

When jobs are subject to a common due window, Yeung, Oğuz and Cheng (2004) propose a heuristic, a branch-and-bound procedure, and a lower bounding scheme based on the dynamic programming algorithm of Weng and Ventura (1996a) for the two-machine flow shop MAD problem. Yeung, Oğuz and Cheng (2004) also claim that the procedure described in Weng and Ventura (1996a) has an error on the recursive relation of the algorithm and show some corrections. The branch-and-bound algorithm is capable of solving instances with up to 25 jobs in size in about 5 minutes, while the heuristic can solve instances of 150 jobs in a few seconds and provide near-optimal solutions. They also demonstrate some polynomially solvable cases.

Chandra, Mehta and Tirupati (2009) subdivide the The m-machine flow shop problem in 3 cases: the due date is such that all jobs are necessarily tardy; the due date is unrestricted; the due date is between the two. A partial characterization of optimal solutions is given for all cases and the authors rely on different heuristics found in literature to obtain approximate solutions. It is also worth noting that they propose a mixed integer programming formulation, however, their CPLEX model can only solve instances up to ten jobs and five machines when the due date is restricted.

İşler, Toklu and Çelik (2012) studied the two-machine flow shop problem with learning and similar assumptions to the model presented in (BISKUP, 1999). Now, the actual processing time of job j in machine i and position r' is given as $p_{ijr'} = p_{ij}r'^{L}$, where L is the learning index. Their approach consists of a mathematical model, a heuristic to determine the starting time of a sequence, a random search, a genetic algorithm, and tabu search. The results are compared and a hybrid strategy (GA and TS) presented the best results.

Table 16 – Summary of algorithms for flow-hop problems

Problem	Complexity	Reference	Algorithm
$F2\|\langle d\rangle, nowait, lrn\|$ $\sum(\alpha E_j + \beta T_j + \gamma d) + \lambda_j(x_j)$	P	Geng, Wang and Bai (2018)	$O(n^3)$
$F2\|\langle d_l, d_r\rangle, nowait, lrn\|$ $\sum(\alpha E_j + \beta T_j + \gamma d_l + \delta(d_r - d_l)) + \rho\lambda_j(x_j)$	P	Shi and Wang (2019)	$O(n^3)$
$F2\|\langle d\rangle, \mathbf{B}, p_j = p\|\sum(E_j + T_j)$	NP-hard	Sung and Min (2001)	*DP*
			$O(n(\bar{c} + \log n))$
$F2\|\langle \hat{d}\rangle, nonprmu\|\sum(E_j + T_j)$	NP-hard	Sarper (1995)	*Enumer.*
			Heuristic
$F2\|\langle \hat{d}\rangle\|\sum(E_j + T_j)$	NP-hard	Sakuraba, Ronconi and Sourd (2009)	*Enumer.*
			Heuristic
$F2\|\langle \hat{d}\rangle, lrn\|\sum(\alpha E_j + \beta T_j)$	NP-hard	İşler, Toklu and Çelik (2012)	*Enumer.*
			Heuristic
$F2\|\langle \hat{d}\rangle, r_j, lrn\|\sum(E_j + T_j)$	NP-hard	Hosseini and Tavakkoli-Moghaddam (2013)	*Enumer.*
			Heuristic
$F2\|\langle \hat{d}_l, \hat{d}_r\rangle\|\sum(E_j + T_j)$	NP-hard	Yeung, Oğuz and Cheng (2004)	*Enumer.*
			Heuristic
$F\|\langle \hat{d}\rangle\|\sum(E_j + T_j)$	NP-hard	Chandra, Mehta and Tirupati (2009)	*Enumer.*
			Heuristic
$F\|\langle \hat{d}\rangle\|\sum(E_j + T_j)$ s.t. WT	NP-hard	Birgin, Ferreira and Ronconi (2020)	*Enumer.*
			Heuristic
$FF\|\langle \hat{d}_l, \hat{d}_r\rangle, size\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Engin and Engin (2018)	*Enumer.*
			Heuristic
$FF\|\langle d_j\rangle\|\sum(E_j + T_j)$ s.t $W.T$	NP-hard	Li et al. (2019)	*Enumer.*
			Heuristic

Hosseini and Tavakkoli-Moghaddam (2013) show an interesting variant of the two-machine problem with learning effects. They assume dynamic arrivals of jobs within a pre-specified time window. The objective is to minimize the deviations of a restrictive common due date as well as the total idle time. A mathematical model, and multi-objective simulated annealing (MOSA) and genetic algorithms are proposed (MOGA). The best results are achieved by MOGA, specially for larger instances.

Shi and Wang (2019) tackle the two-machine no-wait problem with due window assignment, learning effects, and resource allocation, wherein learning effects and resource allocation mean that the processing of a job is a function of its position in a sequence and its amount of resource allocation. Here, the resource allocation is a convex function. The authors show that this problem can be solved in $O(n^3)$. Note that an analogous version of this problem considering the more specific case of a common due date assignment is reported in Geng, Wang and Bai (2018), who show that the optimal solution can also be achieved in $O(n^3)$ time.

In a more complex scenario of a flexible flow shop environment, Engin and Engin (2018) consider the TWET objective function when jobs are subject to a common due window. The window size and location are given parameters. Their solution approach consists of a memetic algorithm in which a global search procedure is accompanied with local search mechanisms. They compare the results of their memetic algotithm with a genetic algorithm and show the superiority in terms of solution quality. Despite developing a sophisticated method, their approach is myopic in the sense of exploring the structural properties of the problem. Table 13 gives the summary of algorithms for flow-shop problems.

Other hybrid flow shop problem is addressed in Li et al. (2019). A mathematical model is given for the problem where P products $P = \{1, 2, ..., n\}$ have to be assembled by J components. Each component, or assembly part J includes Br sub-processes to be performed on m machines located in several production stages following the same technological route. Each stage I has a set of identical parallel machines and each sub-process Br can be processed in any of the machines of each stage I. Furthermore, all components of a product P has a common due date d_J for meeting production requirements. The goal is to minimize the MAD objective function together with the total waiting time (WT). Meta-heuristics were also developed and a modified genetic algorithm produced the best results.

More recently, Birgin, Ferreira and Ronconi (2020) studied the m-machine permutation flow shop for a multi-objective function that includes the MAD from a common due date as well as the sum of waiting times of jobs between machines. They show a mathematical model for the problem, a list scheduling algorithm of complexity $O(n^2m+n^2 \log n)$, and a filtered beam search procedure. A comparison is conducted against the tabu search procedure reported in Chandra, Mehta and Tirupati (2009) and average improvements

are reported specially for larger instances.

Chapter 3

Methodology

According to the literature survey presented in the previous chapter, one problem was chosen and investigated according to a new perspective. As previously evidenced, it turns out that there is no heuristic procedure for this problem. The reminder of this chapter is organized as follows: first, the problem is defined; second, a brief introduction to greedy heuristics is given; third, the problem is subdivided in two phases, which consist of solving the more general parallel machine problem and then focusing on the sequencing aspects on each machine; ultimately, the stochastic local search method is explained. Some useful notations are shown in Table 17.

Table 17 – Notations

Notation	Description
h_l	Restriction factor of the left border of the common due window
h_r	Restriction factor of the right border of the common due window
d_l	Left border of the common due window
d_r	Right border of the common due window
n	Number of jobs
j, k, l	Index of a job
i	Index of a machine
p_j	Processing time of job j
α_j	Earliness penalty of job j per time unit
β_j	Tardiness penalty of job j per time unit
E_j	Earliness of job j
T_j	Tardiness of job j
$s_{[l]}$	Leading idle time, i.e. starting time of the first job
C_j	Completion time of job j
S_j	Starting time of job j
J_E	Set of no-tardy jobs scheduled to finish before or in d_l, $J_E = \{j \mid C_j \leq d_l\}$
J_T	Set of tardy jobs scheduled to begin after or in d_r, $J_T = \{j \mid C_j - p_j \geq d_r\}$
J_W	Set of jobs that start and finish in the due window, $J_W = \{j \mid C_j - p_j \geq d_l \vee C_j \leq d_r\}$
s_l	Left-straddling job, $S_{s_l} < d_l \vee C_{s_l} > d_l$
s_r	Right-straddling job, $S_{s_r} < d_r \vee C_{s_r} > d_r$
s_d	Double-straddling job, $S_{s_d} < d_l \vee C_{s_d} > d_r$

3.1 Problem statement

The problem considers a set of n independent jobs, $N = \{1, 2, ..., n\}$, that need to be processed independently on any of m ($m > 1$) identical parallel machines. The objective is to determine the production sequence of all jobs so as to minimize the total weighted earliness and tardiness penalties. Using the well-known three field classification scheme, the parallel machine scheduling problem (PMSP) can be expressed according to the notations introduced in the last chapter as the triplet $P|\langle \hat{d}_l, \hat{d}_r \rangle| \sum (\alpha_j E_j + \beta_j T_j)$, where E_j and T_j denote the earliness and tardiness of job $j \in N$, respectively. After being processed on the machine, and as a result of scheduling decisions, job j will be assigned a completion time denoted C_j. In other terms, the earliness and tardiness can be expressed as

$$E_j = \max(d_l - C_j, 0) = (d_l - C_j)^+$$

$$T_j = \max(C_j - d_r, 0) = (C_j - d_r)^+$$

It is important to notice that the earliness penalty is non-increasing in C_j. An objective such as the total weighted earliness and tardiness is therefore not regular and α_j and β_j denote the penalty weights corresponding the earliness and tardiness of job j, respectively. For the CDW, let d_l and d_r be the integers indicating the earliest (left boundary) and the latest (right boundary) due dates, respectively. The size and position of the CDW are given by:

$$d_l = \lfloor h_l \cdot (\sum_{j=1}^{n} p_j/m) \rfloor \tag{5}$$

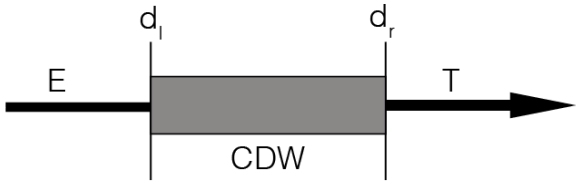
$$d_r = \lfloor h_r \cdot (\sum_{j=1}^{n} p_j/m) \rfloor \tag{6}$$

Here, h_l and h_r are given parameters for defining d_l and d_r, respectively. While the parameters h_l and h_r could be specified by any value, it is crucial to define their ranges as $0 < h_l < h_r < 1$, so that the CDW becomes restricted by $0 < d_l < d_r < (\sum_{j=1}^{n} p_j/m)$. This configuration allows to avoid some trivial cases, for example, when $h_l = 0$ and $h_r = 1$, the CDW size is equal to $(\sum_{j=1}^{n} p_j/m)$ and all the jobs can be completed within the CDW. In addition, if $h_l = h_r$, then the CDW reduce to a time instant and could therefore be considered as a common due date. Thus, eq. 5 and 6 permit a good definition of the CDW and figure 2 depicts it. Other important assumptions are given as follows:

- ❏ All jobs are independent and assumed to be ready at time zero.
- ❏ The machines can process only one job at a time and must process jobs without any interruption from the start of processing the first job to the completion of the last job.

- ❏ No pre-emptions are allowed.
- ❏ Machine breakdowns are not considered.
- ❏ The CDW has constant size and position.

Figure 2 – Common due window



As discussed in Chapter 2, two cases of the problem can be distinguished. The first case implies that the left boundary of the due window (d_l) is large enough to accommodate as many jobs as possible. In the second case, however, it restricts the amount of jobs that can be processed before the interval $[0, d_l]$. Note that, any algorithm that solves both cases in the parallel machine environment has two main decisions to address: how to assign jobs to machines and how to sequence the jobs assigned to each machine (ALVAREZ-VALDES; TAMARIT; VILLA, 2015). The solution procedure reported in this study answers these questions with a family of heuristics containing the following structure:

- ❏ Five priority rules for assigning jobs onto machines are proposed. These rules are based on job characteristics, which are inherent to the problem (processing times, earliness, and tardiness penalties).
- ❏ A greedy constructive algorithm is then applied to generate a feasible initial sequence on each machine. Since the initial assignment may be myopic in relation to the the charge of jobs distributed to each machine, a local search procedure is incorporated to refine the preliminary solution.
- ❏ Once an initial solution is obtained with a better distribution of jobs, sequencing algorithms are applied for each machine subproblem separately. There are only two algorithms reported in literature for the single machine case: the GH and RGH heuristics (BISKUP; FELDMANN, 2005; YING; LIN; LU, 2017). The RGH was chosen since it is an improved version of the GH. In addition, a sequential exchange approach (SEA) is proposed. It is worth mentioning that both algorithms do not require a sequence to start at time zero.
- ❏ To produce better quality solutions, a simple iterated greedy (IG) algorithm is implemented. It is a stochastic local search algorithm that iterates through the application of construction heuristics using the repeated execution of two main phases, the partial destruction of a complete candidate solution and a subsequent reconstruction of a complete candidate solution.

3.2 Greedy construction heuristics

The proposed initial assignment algorithms are greedy strategies in some sense. It implies that they build candidate solutions step by step, starting from an empty solution. At each step, a new component is added to a current partial solution. These steps are repeated until a complete solution is obtained. In general, there is a heuristic function that evaluates the gain obtained by inserting this new element into the partial candidate solution. Of course, if more than one component has the same best heuristic value, a tiebraking criterion is applied to decide which component should be incorporated; the tiebreaking is usually performed at random or by a secondary heuristic function. Algorithm 1 represents the outline of a greedy heuristic.

Algorithm 1 Greedy constructive heuristic

```
π ← ∅
while π is not a complete solution do
  Choose a best rated solution component c
  π ← π + c
end while
end
```

Greedy heuristics are commonly chosen to tackle combinatorial optimization problems for several reasons. First, they frequently generate solutions that are better than those generated uniformly at random or by a randomized but heuristically biased construction. Second, in general they are fast procedures. Third, these algorithms are often used to seed local search methods such as iterative improvement algorithms; more sophisticated methods, such as tabu search and simulated annealing; or population-based metaheuristics such as genetic, and memetic algorithms, and other evolutionary strategies. In the latter case, normally some population members are generated by greedy constructive methods, while others may be generated entirely at random. Seeding perturbative local search methods with solutions from a greedy heuristic can lead to advantages such as improved quality of local optima, faster identification of local optima, and a better trade-off between computation times and solution quality, that is, better anytime behavior. Fourth, one can prove guarantees on the quality of optimal solutions, establishing a worst-case error bound. Finally, they build the basis for a number of other methods, such as GRASP, ant-colony optimization (ACO), and others (STÜTZLE; RUIZ, 2018).

One simple way to improve over the generation of a single greedy solution is, in some cases, the repeated application of a greedy heuristic to generate a variety of possible candidate solutions and then choose the best one. It is important to mention that, in these cases, repetition only makes practical sense if in the construction process different solutions can be generated.

Obviously, the repeated construction of solutions also has implicit disadvantages. Constructing a full solution may require additional computational resources as the initial

construction steps demand a large amount of computation when compared to later construction steps. Moreover, no information is taken from previous solutions (STÜTZLE; RUIZ, 2018). To mitigate these problems and guarantee the best possible solution, the IG proposed in this study permits to invest, in principle, arbitrary computing times to generate different solutions by constructive heuristics.

3.3 Solving the parallel machine problem

For solving the parallel machine problem, one can rely on the branch-and-bound algorithm proposed by Chen and Lee (2002). Their algorithm is based on a column generation approach in which the problem is first formulated as a set partitioning type and then in each branch-and-bound iteration of the linear relaxation of this formulation is solved by a standard column generation procedure. However, their model is rather complex to implement and computationally expensive. Therefore, the proposed algorithmic scheme combines heuristic priority rules with other neighborhood search strategies in order to produce good quality solutions with reduced computing times. The initial solution process is divided into four different phases:

- **Phase 1:** Generate an initial solution by assigning jobs to machines according to five different heuristic criteria.
- **Phase 2:** Local search to study whether it is possible to find a better distribution of the jobs onto machines.
- **Phase 3:** Once a better distribution of jobs is achieved, two sequencing strategies are applied to solve each machine subproblem (RGH, SEA).
- **Phase 4:** An IG algorithm is incorporated to the best-so-far procedure to diversify the search space and look for a better local optima.

3.3.1 Initial assignment

The initial assignment considers the whole list of jobs. This list is ordered according to the priority criteria for the similar problem of identical parallel machine scheduling against a common due date. Alvarez-Valdes, Tamarit and Villa (2015) tested several other assignment rules, however, the following rules appear in the list because they obtained the best results:

- **Rule 1:** Non-increasing $\beta_j p_j$.
- **Rule 2:** Non-increasing $\beta_j p_j / \alpha_j$.
- **Rule 3:** Non-increasing p_j.

❑ **Rule 4:** Non-increasing β_j.

❑ **Rule 5:** Non-decreasing p_j/β_j

Once the whole list of jobs is ordered, it is time to distribute the jobs onto machines. The first $m-1$ jobs are assigned to the first $m-1$ machines. From then on, the assignment algorithm proceeds by taking the next job in the list and calculating the machine and the position in which it produces the minimum cost increase. In addition, it also considers the next job in the list. The calculations are made for both of them and the job producing the minimum cost is assigned to the machine and position in which this minimum cost was attained. The other job will then become the first in the list and will compete with the next one. The pseudocode of the initial assignment strategy is given in Algorithm 2.

Algorithm 2 Initial assignment strategy

```
job_list ← {1, 2, ..., n}, flag ← true
Order job_list according to one of the priority rules
while job_list ≠ ∅ do
  if flag = true then
    Assign the first m − 1 jobs to the first m − 1 machines
    flag ← false
  else
    Let job k be the head of job_list
    Insert jobs k and k + 1 in every possible position in each machine
    Compute global costs
    Index the job that produced the minimum cost increase to the machine and position
    this cost was attained
    Return the worst job to the head of job_list
  end if
end while
end
```

3.3.2 Local search

The local search (LS) procedure was designed to exchange jobs between two machines that have different job charges. In other words, since the initial assignment do not consider the balance between earliness and tardiness penalties as well as processing times, LS looks for a decrease in the cost function by moving jobs from a machine to another. Algorithm 3 represents the pseudocode of LS.

The procedure starts by ordering the machines by non-increasing costs. Once the machines are ordered, for each machine i it computes the total processing time D_i^T, the sum of earliness penalties α_i^T, and the sum of tardiness penalties β_i^T. Then, it compares the machine with a higher cost i with the next machine with a lower cost $i+1$ and checks the insertion of job j into $i+1$ if at least one of the following conditions hold:

$$D_i^T > D_{i+1}^T + p_j \vee \alpha_i^T > \alpha_{i+1}^T + \alpha_j \vee \beta_i^T > \beta_{i+1}^T + \beta_j$$

If some of these conditions hold, it tries to move job j from machine m to machine $m+1$ by checking the insertion of j in every possible position of machine $m+1$. If a global improvement is achieved, job j is indexed into the best possible position of $m+1$. The procedure is repeated until all machines have been checked.

Algorithm 3 Local search (LS)

```
machines ← {1, 2, ..., m}, imporve ← false
Order machines by non-increasing costs
for i = 1 to m − 1 do
  while improve = true do
    Compute D_i^T, α_i^T, β_i^T, D_{i+1}^T, α_{i+1}^T, β_{i+1}^T
    for all jobs j ∈ m do
      if D_i^T > D_{i+1}^T + p_j ∨ α_i^T > α_{i+1}^T + α_j ∨ β_i^T > β_{i+1}^T + β_j then
        Try to move job j from i to i + 1 by inserting it in every possible position of
        i + 1.
        Compute the global costs
        if obj(π_new) < obj(π) then
          Move job j from i to i + 1 to its best possible position.
          improve ← true
        else
          Keep π unchanged
          improve ← false
        end if
      end if
    end for
  end while
end for
end
```

3.3.3 Solving the single-machine subproblems

Once the jobs are distributed to machines, it is necessary to find better sequences on each machine. For this, two strategies are adopted: the RGH heuristic and SEA. Both deterministic methods are used to order the sequences on each machine. This is the third phase of the solution procedure. Hereafter, the nomenclature adopted to differentiate each solution procedure is RN-RGH and RN-SEA. Note that the nomenclature is followed by a number from 1 to 5, which distinguishes the 5 assignment rules. The next two subsection describes these solutions procedures in detail.

3.3.3.1 RGH heuristic

As previously mentioned, the RGH heuristic is an improved version of the GH heuristic. It was originally designed to gain a reasonable starting solution for meta-heuristic approaches for the single machine scheduling problem with a common due window. The underlying idea of this simple heuristic is to assign those jobs to set J_W which have high earliness and tardiness penalties relative to their processing times. Afterwards from the remaining jobs, those which possess a high tardiness penalty relative to the earliness penalty are assigned to set J_E. Algorithm 4 shows the sequencing strategy outlined by the RGH heuristic. Note that the procedure is repeated for each machine until all machines have been sequenced. The improvements made by RGH heuristic in relation to GH heuristic relies on the fact that the GH does not take into account the cases where a left-straddling job s_l may occur. To correct this weakness, Ying, Lin and Lu (2017) use three different rules to produce a furthest forward-shift solution (the one that starts at time zero), the forward-shift solution of the straddling job and the backward-shift solution of the straddling job. Of course, shifting the schedule may require to rearrange the schedule according to the v-shape property.

3.3.3.2 Sequential exchange approach

SEA was originally proposed for the single machine problem with a common due date (LIN; CHOU; YING, 2007). Here, a new heuristic is shown to schedule each machine against a common due window. Since there are three sets of jobs that constitute a solution for the problem (J_E, J_W, J_T), SEA swaps or moves jobs between these sets systematically to derive a near-optimal solution. For the sake of simplicity, only two sets are considered since jobs in J_W may be sequenced in an arbitrary order. Now, let $J_{E'}$ denote the set of no-tardy jobs including the ones that start and finish in the due window ($J_{E'} = J_E \cup J_W$). Possible straddling jobs (s_l, s_r, and s_d) are also in $J_{E'}$ since they have to start processing before the right boundary of the due window.

For each machine, the jobs in $J_{E'}$ are selected one by one according to the sequence obtained as a result of phases one and two. This step is called forward exchange. The selected job $j \in J_{E'}$ is paired with all jobs in J_T to identify the swap which would lead to the largest improvement of the cost function value. Furthermore, job $k \in J_{E'}$ may move to J_T if this move leads to a greater improvement of the cost function than swapping jobs. If swapping the jth job in $J_{E'}$ with the lth job in J_T causes the best gain in the objective function value than moving, the jth job in $J_{E'}$ and the lth job in J_T are swapped, while the original sequences in $J_{E'}$ and J_T are maintained. On the other hand, if moving the kth job in $J_{E'}$ to J_T results in a best improvement than swapping, the kth job moves to J_T. After trying to swap or move all jobs in $J_{E'}$ to J_T, the best move is performed and the jobs are ordered according to the v-shape property again.

Algorithm 4 RN-RGH heuristic

$machines \leftarrow \{1, 2, ..., m\}$
for $i = 1$ to m **do**
$J_T \leftarrow \{1, 2, ..., n_i\}$, $J_W \leftarrow \varnothing$, $J_E \leftarrow \varnothing$ // n_i *is the number of jobs assigned to machine* i.
Set $gap1 = (d_r - d_l) - \sum_{j \in J_W} p_j$
while $gap1 > \min\{p_j | j \in J_T\}$ **do**
From all jobs $j \in J_T$ with $p_j \leq gap1$, select the job with the highest ratio $\min(\alpha_j, \beta_j)/p_j$ among all jobs in J_T (named job k), then sequentially assign it to J_W. In case of a tie, select the job with the largest value of $\min(\alpha_j, \beta_j)$. Delete job k from J_T.
Update $gap1$.
end while
Set $gap2 = d_l + gap1$.
Sequence all jobs in J_T in non-increasing order of β_j/p_j. In case of a tie, sequence the jobs in non-increasing order of the ratio β_j/α_j. If there is still the case of a tie, sequence the jobs in non-increasing order of p_j. Set $\pi = (J_E \cup J_W \cup J_T)$ and $s_{[l]} = gap2$.
while $gap2 > \min\{p_j | j \in J_T\}$ **do**
For all jobs $j \in J_T$ with $p_j \leq gap2$ that are not yet tested. Try to move the job with the highest ratio β_j/α_j (named job k) to J_E according to the v-shape property. In case of a tie, move the job with the smallest ratio α_j/p_j to J_E. Set $\pi_{new} = (J_E \cup J_W \cup J_T)$ with $s_{[l]} = gap2 - p_k$.
if $\text{obj}(\pi_{new}) < \text{obj}(\pi)$ **then**
$\pi \leftarrow \pi_{new}$.
else
Keep π unchanged.
end if
if All jobs have been tested **then**
Break.
end if
end while
Set $s_{[l]} = 0$ and obtain the furthest forward-shift solution π^{f1}_{new} and check whether a left-straddling job s_l occurs. Rearrange jobs v-shape.
if s_l occurs **then**
Set $s_{[l]} = \max\{0, gap2 - gap1\}$ to obtain the forward-shift solution of the straddling job π^{f2}_{new}. Rearrange jobs v-shape.
Set $s_{[l]} = gap2 + gap3$ to obtain the backward-shift solution of the straddling job π^{b}_{new}. Where $gap3 = p_{s_l} - gap1$. Rearrange jobs v-shape.
Output the best solution among π, π^{f1}_{new}, π^{f2}_{new}, and π^{b}_{new}.
else
Output the best solution among π, and π^{f1}_{new}.
end if
end for
end

Algorithm 5 RN-SEA heuristic

$machines \leftarrow \{1, 2, ..., m\}$
for $i = 1$ to m **do**
  Forward exchange:
  while $improve = true$ **do**
    $improve \leftarrow false$
    for all jobs $j \in J_{E'}$ **do**
      Swap job j with all jobs $l \in J_T$ and calculate obj(π_s) with start time $s_{[l]} = 0$.
      Move job j to all possible positions of J_T and calculate obj(π_m) with start time $s_{[l]} = 0$.
      if obj(π_s) < obj(π) or obj(π_m) < obj(π) **then**
        Save the best improvement. Keep π unchanged.
        $improve \leftarrow true$
      end if
    end for
    Swap or move the job that led to a greater improvement in the objective function value.
    Update π and rearrange it according to the v-shape property.
  end while
  Backward exchange:
  while $improve = true$ **do**
    $improve \leftarrow false$
    for all jobs $l \in J_T$ **do**
      Swap job l with all jobs $j \in J_{E'}$ and calculate obj(π_s) with start time $s_{[l]} = 0$.
      Move job l to all possible positions of $J_{E'}$ and calculate obj(π_m) with start time $s_{[l]} = 0$.
      if obj(π_s) < obj(π) or obj(π_m) < obj(π) **then**
        Save the best improvement. Keep π unchanged.
        $improve \leftarrow true$
      end if
    end for
    Swap or move the job that led to a greater improvement in the objective function value.
    Update π and rearrange it according to the v-shape property.
  end while
  Setting the start time:
  Keep the solution with $s_{[l]} = 0$. Identify whether a left or/and a right straddling job exists. Set the start time as $s_{[l]} = d_l - S_{s_l}$ and/or $s_{[l]} = d_r - S_{s_r}$ and rearrange the schedule according to the v-shape property. Output the sequence with the best objective function value.
end for
end

The next step is called the backward exchange, which is similar to the forward exchange. However, jobs in J_T are selected one by one according to the sequence obtained from the previous step, and jobs are swapped with jobs in $J_{E'}$ to find the best improvement of the objective function value. Conversely, jobs in J_T are moved to $J_{E'}$ if the best improvement if a better gain is attained. After examining all the jobs in J_T to be swapped or moved to $J_{E'}$, the best move is performed and the final schedule is sequenced according to the v-shape property once again.

Since the common due windows are restricted in relation to the total processing time, the start time of the first job to be scheduled is set to be zero. Evidently, for instances where the left boundary (d_l) has a larger restrictive factor (h_l), it may be the case that possible straddling jobs should be eliminated to obtain better objective function values, therefore, the procedure identifies them and produces backward-shift solutions of right and left straddling jobs. The start time is calculated as $s_{[l]} = d_l - S_{s_l}$ and $s_{[l]} = d_r - S_{s_r}$. Of course, jobs should be rearranged according to the v-shape property and the solution with the best objective function value is kept.

3.3.4 Iterated greedy

To put it briefly, IG generates a sequence of solutions by iterating over greedy constructive heuristics (see section 3.2) using two important phases: destruction and construction. During the destruction phase, some solution components are removed from a previously constructed complete candidate solution. The construction procedure then applies a greedy constructive heuristic to reconstruct a complete candidate solution. Once this new solution is constructed, an acceptance criterion is used to select a new incumbent solution. The algorithm iterates over these steps until a stopping condition is reached. Note that, IG is closed related to iterated local search (ILS), however, instead of iterating over a local search, it iterates over construction heuristics (RUIZ; STÜTZLE, 2007). Figure 3 depicts a hypothetical iteration of the IG algorithm in one of the machines and Algorithm 6 outlines the step by step procedure.

3.3.4.1 Destruction and construction

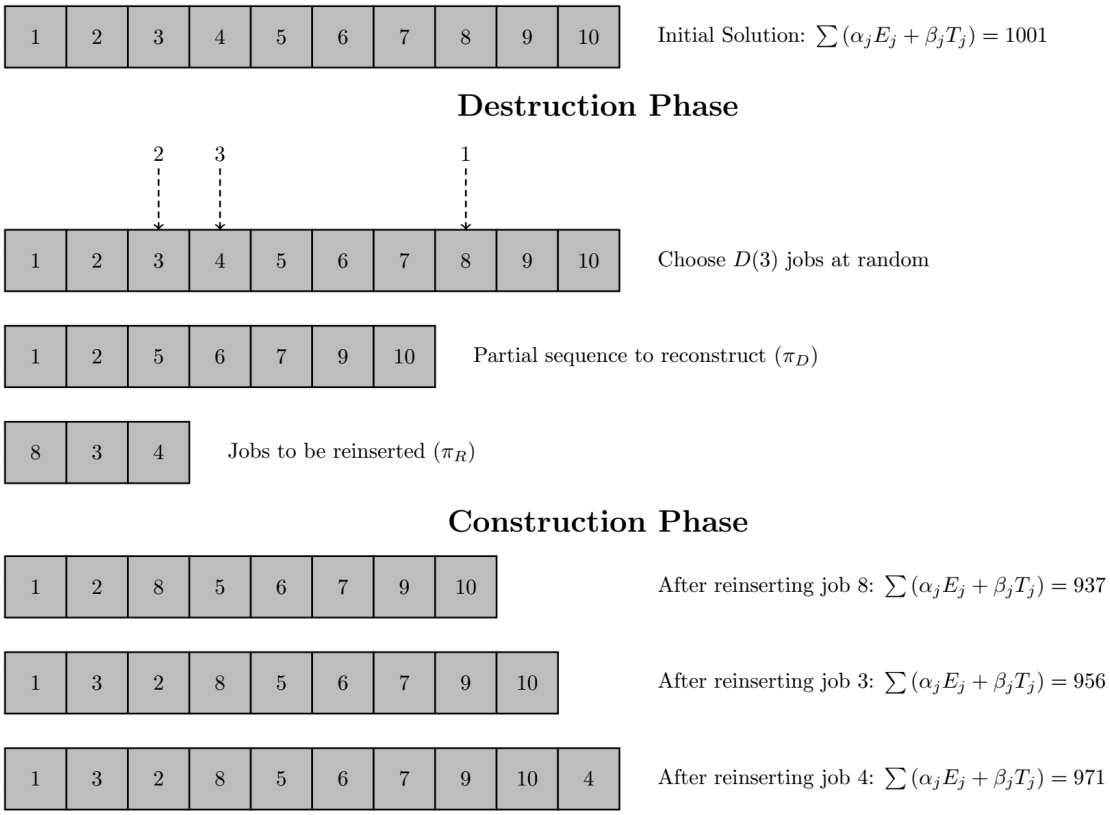
Two central procedures in any IG algorithm are the destruction and construction procedures. The destruction procedure is applied to a permutation π of n jobs and it chooses randomly and without repetition D jobs. These jobs are then removed from π in the order in which they were chosen. As a result, there are two subsequences, the first is a partial sequence π_D with $n - D$ jobs and another sequence of D jobs which is denoted as π_R. It is worth noting that π_R contains the jobs that need to be reinserted into π_D in the order in which they were removed from π. Here, the destruction and construction procedures are repeated to every machine to build a new feasible solution.

Once a complete feasible solution is generated, the LS procedure described in Algorithm 3 is applied.

3.3.4.2 Acceptance criterion

After the local search phase, an acceptance criterion is applied to determine whether the new generated solution π^*_{new} is accepted or not as the incumbent solution π^* for the next iteration. The simplest way to establish an acceptance criteria is to accept only better solutions. However, an IG algorithm using such strategy may lead to premature convergence due to insufficient diversification (RUIZ; STÜTZLE, 2007). Other authors prefer to choose a fixed probability of accepting worst solutions, which is a parameter to be determined (RIBAS; COMPANYS; TORT-MARTORELL, 2011; RODRIGUEZ et al., 2013; ARROYO; LEUNG, 2017; RIBAS; COMPANYS; TORT-MARTORELL, 2019). Other authors opt for a simulated annealing acceptance criterion where the acceptance probability is based on a control parameter called temperature. This temperature may be constant (RUIZ; STÜTZLE, 2007; YING; LIN; HUANG, 2009; LIN; LU; YING, 2011) or variable (LIN et al., 2011; KANG; HE; WEI, 2013). In this study, we adopt the simulated annealing acceptance criterion described in Ruiz and Stützle (2007) with a constant temperature. The notation *rand* of the acceptance criterion depicted in Algorithm 6 refers to a random generated number taken from the interval $[0, 1]$.

Figure 3 – Example of one iteration of the proposed IG heuristic



3.3.4.3 Stopping conditions

The stopping condition commonly involves a fixed number of iterations, a depleted execution time, or the detection of algorithmic stagnation. In this study, only one stopping condition is used to force the IG algorithm to be terminated, which consists of the depleted execution time. In other words, once a certain amount of CPU time is reached, the IG algorithm is stopped.

Algorithm 6 Iterated Greedy (IG)

$\pi \leftarrow Initial\_Assignment(\pi)$
$\pi \leftarrow Local\_Search(\pi)$
$\pi \leftarrow Best\_Heuristic(\pi)$
$\pi_{best} \leftarrow \pi$
while Stopping condition is not satisfied **do**
  Destruction Phase:
  $\pi' \leftarrow \pi$
  for $j = 1$ to D **do**
    $\pi' \leftarrow$ remove one job at random from π' and insert it in π'_R.
  end for
  Construction Phase:
  for $j = 1$ to D **do**
    $\pi' \leftarrow$ best permutation obtained by inserting job $j \in \pi'_R$ in all possible positions of π' according to the v-shape property.
  end for
  Local Search:
  $\pi'' \leftarrow Local\_Search(\pi')$
  if $\text{obj}(\pi'') < \text{obj}(\pi)$ **then**
    $\pi \leftarrow \pi''$
    if $\text{obj}(\pi'') < \text{obj}(\pi_{best})$ **then**
      $\pi_{best} \leftarrow \pi$
    end if
    Acceptance criterion:
  else if $(rand \leq \exp\{-(\text{obj}(\pi'') - \text{obj}(\pi))/Temp\})$ **then**
    $\pi \leftarrow \pi''$
  end if
end while
return π_{best}
end

Chapter 4

Numerical experiments

In this chapter, a numerical analysis of the proposed methodology is conducted. Since there is no heuristic procedure for the problem under consideration, this analysis is focused on the performance assessment between the algorithms reported in the last chapter. The algorithms were coded in C++ programming language using Visual Studio$^{(TM)}$ 2019 on Windows$^{(TM)}$ 10 and ran on a personal computer (PC) with an Intel® Core$^{(TM)}$i5 (3.7GHz) and 16 GB of RAM. The benchmark instances are available at the well-known O.R Library (BEASLEY, 1990). The short Pascal generator as well as the problem instances were proposed for the analogous common due date single machine problem reported by Biskup and Feldmann (2001). Here, for the window configuration factors h_l and h_r, it was decided to set three different values of h_l ($h_l \in \{0.1, 0.2, 0.3\}$) and four different values for h_r ($h_r \in \{0.2, 0.3, 0.4, 0.5\}$), which culminate in five possible configurations of due windows (BISKUP; FELDMANN, 2005) that are calculated as specified in Chapter 2 according to the processing time and machine configuration of each instance. Furthermore, three possible machine environments are tested for ten different instances, which, in turn, have five different sizes. This way, a total of $5 \times 3 \times 10 \times 5 = 750$ instances are solved by each algorithm. The characteristics of the instance set is depicted in Table 18.

Table 18 – Instance configuration

Factors	Levels
n	20, 50, 100, 200, 500
m	2, 4, 6
(h_l, h_r)	(0.1, 0.2), (0.1, 0.3), (0.2, 0.5), (0.3, 0.4), (0.3, 0.5)
α_j	$U[1, 10]$
β_j	$U[1, 15]$
p_j	$U[1, 20]$

In the following sections, the design of experiments for the parameter settings of the IG algorithm is presented, and then the performance evaluation of the IG and the other heuristics is reported, which is based on two central measures: the quality of the solutions,

which is the relative percentage deviation (RPD) from the best solution found and the computational effort to obtain theses solutions, which is determined by the CPU time. Again, it is important to notice that the IG receives the solution of the best performing heuristic as an input. Ultimately, a statistical analysis is executed to determine whether there is a significant difference on the results obtained by each algorithm.

4.1 Parameter setting

The developed IG depends on three parameters: the number of jobs to be removed from a solution D, the temperature $Temp$ for the simulated annealing acceptance criterion, and the stopping condition. D is the number of jobs to be removed from a machine, which is given as $D = \lceil d' * n_m \rceil$, where $0 < d' < 1$ and n_m is the number of jobs assigned to a given machine. The stop criterion is given as the elapsed CPU time equal to $t * n$ seconds $(0 < t < 1)$. Of course, in order to obtain a better performance of the IG, it is desirable to determine the proper values of the parameters d', $Temp$, and t.

Table 19 – Full factorial experiment

Algorithm	d'	$Temp$
Alg 1	0.01	5
Alg 2	0.01	10
Alg 3	0.01	20
Alg 4	0.01	50
Alg 5	0.05	5
Alg 6	0.05	10
Alg 7	0.05	20
Alg 8	0.05	50
Alg 9	0.10	5
Alg 10	0.10	10
Alg 11	0.10	20
Alg 12	0.10	50
Alg 13	0.20	5
Alg 14	0.20	10
Alg 15	0.20	20
Alg 16	0.20	50

To avoid the risks of over-fitted results, it was selected 5 out of 10 instances of each size totally at random. For this, 5 non-repeated numbers between 0 to 10 were generated and the respective instances were taken from the test set, e.g., each of the 10 instances of size $n = 20$ were matched with random indexes and taken as samples to determine the parameter levels. This way a total of $5 \times 3 \times 5 \times 5 = 375$ instances were solved for different parameter configurations. With a fixed CPU time equal to $0.10 * n$ seconds $(t = 0.10)$, a full factorial experiment with d' and $Temp$ was conducted and each parameter is tested

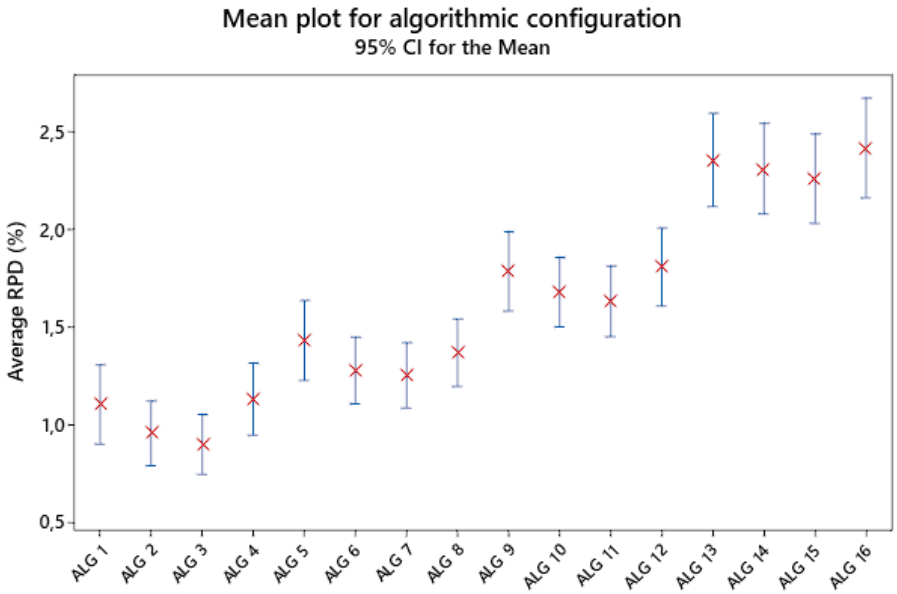
at four levels resulting in a total of 16 different algorithms. Each configuration is reported in Table 19.

After setting each version of the IG algorithm to solve each instance 5 times, the quality of the solutions is measured by means of the RPD, which is computed according to the following equation:

$$\text{RPD}(\%) = \frac{\text{obj}^{\text{avg}} - \text{obj}^{\text{best}}}{\text{obj}^{\text{best}}} \times 100$$

Where obj^{avg} stands for the average objective function value of 5 runs of a specific version of the IG and obj^{best} refers to the minimum sum of penalties among all algorithms. The results are then evaluated according to the average RPD values. For better visualization, Figure 4 represents the mean plot of average RPD values with 95% confidence interval of different parameter levels.

Figure 4 – Mean plot for algorithmic configuration of parameters *Temp* and d'



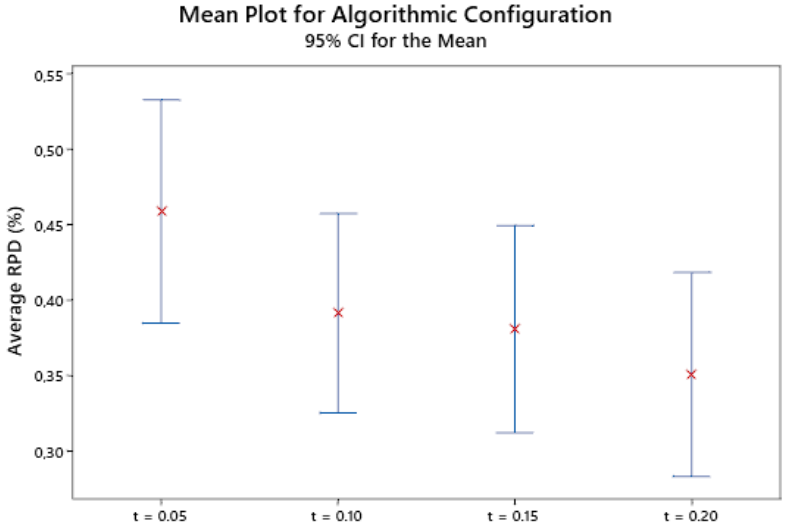
In Figure 4 it is possible to identify that smaller values of d' lead to better results since smaller values of RPD represent better solutions. Note that there is a clear difference between groups with distinct levels of d', e.g., algorithms 1, 2, 3, 4, which have an equivalent value of $d' = 0.01$ perform better in terms of solution quality than the other three groups consisting of algorithms 5, 6, 7, and 8 ($d' = 0.05$), algorithms 9, 10, 11, and 12 ($d' = 0.10$), and algorithms 13, 14, 15, and 16 ($d' = 0.20$). It is also worth mentioning that IG algorithms of these groups present confidence intervals of average RPD values that do not seem to overlap, which may be considered as a strong evidence that the group with $d' = 0.01$ has a more consistent performance in comparison with the others. For this reason, d' was chosen to be 0.01.

To determine $Temp$ it is fundamental to establish a range that guarantees the convergence of the IG algorithm. At a first moment, $Temp$ was set following the equation suggested by Ruiz and Stützle (2007):

$$Temp = T \times \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij}}{n \times m \times 10}$$

Note that T is usually a real number in the interval $[0, 1]$ that needs to be adjusted. The equation, however, did not seem to produce good results after a preliminary analysis. Taking instance *sch*.20.4 from the test set presented by Biskup and Feldmann (2001) as an example, the sum of processing times is equal to 230. Now, supposing that $m = 2$ and $T = 1$, $Temp$ would have a value of approximately 0.003. After some empirical testing, it was observed that such a small value for $Temp$ yielded poor results. To deal with this issue, an empirical analysis was conducted for values of $Temp$ starting at 3000 and then decreased to obtain a reasonable range. This empirical analysis included different due-window configurations, machine numbers as well as instance sizes. After some testing, it was identified that better results were achieved for values from 5 to 50. Figure 4 shows the behavior of the different values of $Temp$. It is important to notice that algorithms with $Temp = 20$ presented a lower average RPD in comparison to the other levels, however, the confidence intervals for these averages overlap, which might indicate the proximity between the solution quality achieved by the algorithms. In addition, when analyzing the behavior of the confidence intervals, it is possible to observe a valley-shaped relation, that is, the average RPD starts to decrease from $Temp = 5$ to $Temp = 20$ when it starts to increase again. This fact corroborates with the notion that an ideal value for $Temp$ would lay between 10 and 50. For the sake of simplicity, the value of $Temp$ was set to 20 since it presented the lowest average RPD. Stated in other terms, the levels for $Temp$ and d' are set according to *Alg* 3 (see Table 19).

Figure 5 – Mean plot for algorithmic configuration of parameter t



After calibrating the parameters d' and $Temp$, the IG algorithm was set to run with other CPU times as stopping criterion. Four different values for the parameter t are compared and the CPU times are given as: $0.05 * n$, $0.10 * n$, $0.15 * n$, and $0.20 * n$. Figure 5 shows the mean plot for RPD values at 95% confidence intervals. It is possible to verify that the confidence intervals for the average RPD seems to overlap for all possible levels of t. This fact may indicate that the IG algorithm converges quickly. Note that the smallest RPD value is attained for $t = 0.20$, therefore, this valued is chosen in the experiments as the stop criterion.

4.2 Experimental evaluation

To demonstrate the performance of the heuristics, the average RPD values are compared based on three factors that modify the characteristics of the problem: instance size; window configuration; and number of machines. Table 20 summarizes the computational results of all proposed methods in terms of the solution quality for different instance sizes and the best results are marked in bold. There is a clear difference in the performance measures of the 11 heuristics, which stems from three major aspects: the initial priority rule used to order jobs before assigning them to machines; the sequencing strategy adopted on each machine; and the application of the stochastic local search (IG algorithm).

Table 20 – Average RPD values for each size of test instances

Algorithm	$n = 20$	$n = 50$	$n = 100$	$n = 200$	$n = 500$
RN-RGH-1	21.86	12.37	8.29	6.24	5.70
RN-RGH-2	21.08	14.69	11.87	10.08	10.75
RN-RGH-3	21.13	13.31	9.20	6.23	5.07
RN-RGH-4	17.45	12.09	8.12	6.01	5.06
RN-RGH-5	15.82	13.70	10.44	8.51	9.11
RN-SEA-1	3.87	2.98	2.10	1.51	1.76
RN-SEA-2	4.68	4.77	4.57	4.79	5.96
RN-SEA-3	4.74	4.42	3.56	1.85	1.08
RN-SEA-4	2.70	1.03	1.01	0.75	0.77
RN-SEA-5	5.69	2.58	1.65	2.00	3.05
IG_{SEA-4}	**1.03**	**0.17**	**0.10**	**0.12**	**0.24**

Clearly, the best performance is achieved for rule 4 (non-increasing β_j). When comparing the results for the same sequencing strategies, that is, between the 5 rules using the same sequencing algorithm (RN-RGH or RN-SEA), this rule outperforms the others for almost all instance sizes, except when combined with RN-RGH for solving instances with $n = 20$. Such an effect may be explained due to the nature of the data of the test set. Note that the individual tardiness penalty is allowed to be greater than the earliness penalty (refer to Table 18). In most practical contexts, contractual penalties incurred

by a tardy job is usually higher than the cost of maintaining an early job in the inventory. Other possible explanation of the success of this rule relies on the fact that due windows are tight in relation to the sum of processing times. Following this rationale, a larger number of jobs should be scheduled in J_T, therefore, the amount of tardiness penalties is expected to be greater than the amount of earliness penalties. In terms of the improvements methods applied to sequence jobs in each machine, RN-SEA presented a consistently better performance. The reason for this behaviour relates to the fact that RN-RGH might lead to poor results if there are many jobs with $\beta_j \gg \alpha_j$, where α_j is near to one and d_l is relatively small: in this situation, probably not all jobs with high tardiness penalties can be processed prior to the common due window and hence accumulate higher tardiness costs. Note that the good performance of RN-SEA also corroborates with the results previously reported in Lin, Chou and Ying (2007), when the application of a similar strategy to solve the analogous common due date single-machine problem presented better results in relation to several *ad hoc* heuristics and metaheuristics. Finally, the stochastic local search combined with the best performing strategy ($IG_{SEA\text{-}4}$) improved the average RPD for instances of all sizes. For a better visualization, Figure 6 depicts the estimated marginal means according to instance size.

Figure 6 – Estimated marginal means of RPD values according to instance size

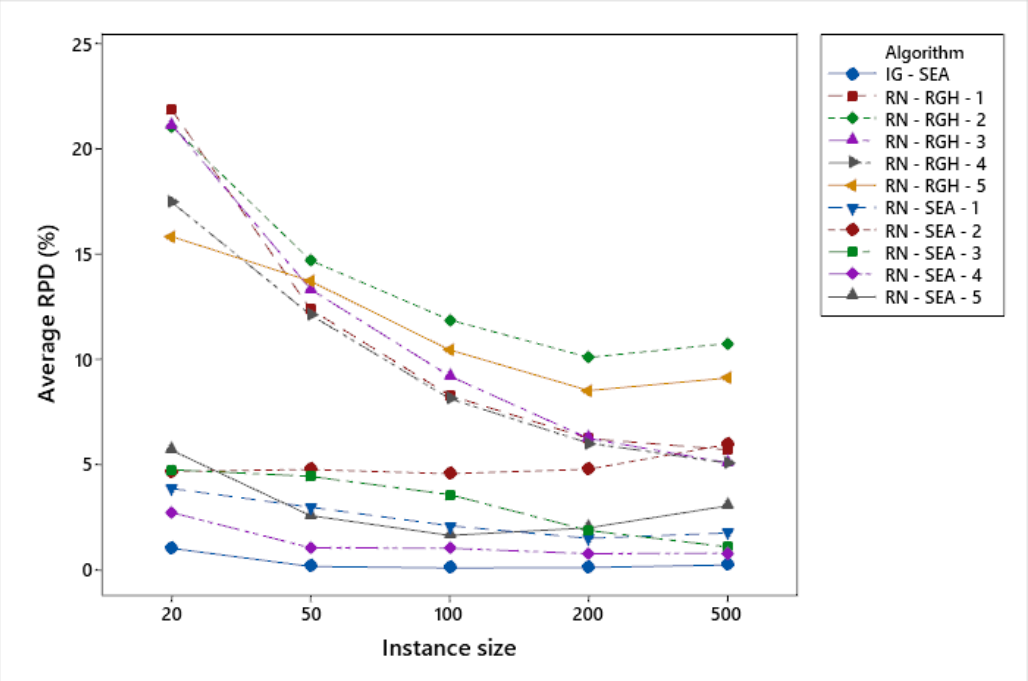
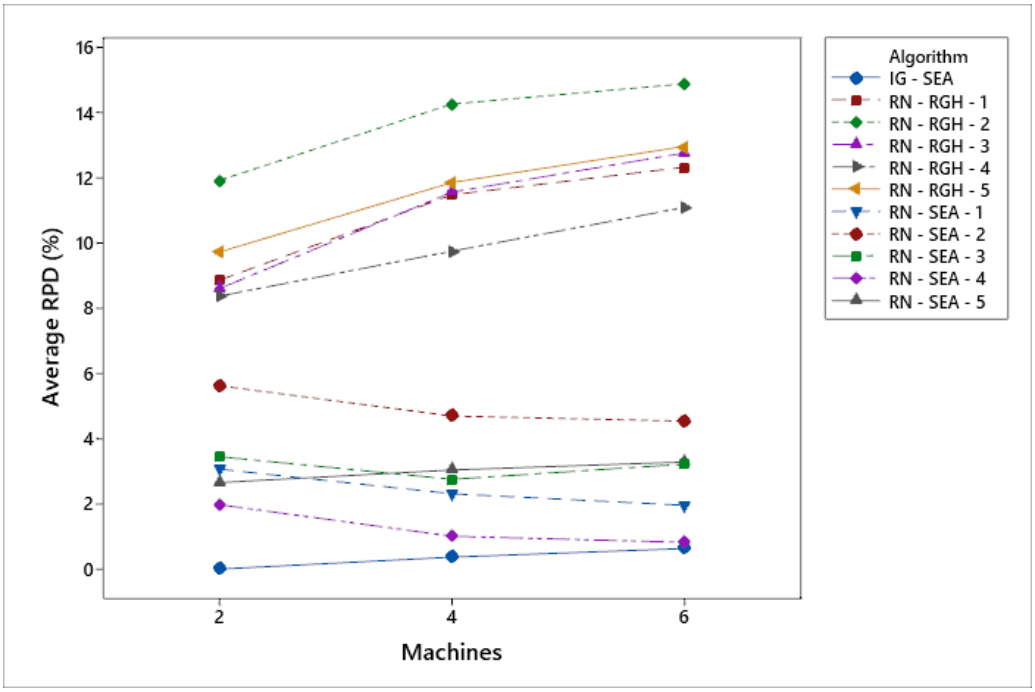


Figure 6 also confirms that RN-SEA-4 is capable of exploring the search space in an efficient way. Note that, despite the fact that the IG algorithm finds better solutions in all cases, the RPD difference in relation to RN-SEA-4 are relatively small, which support the idea that it is possible to find good solutions with a constructive heuristic that is simple and do not require any parameter setting. A similar behaviour is seen in relation to the number of machines as illustrated in Figure 7.

Figure 7 – Estimated marginal means of RPD values according to the number of machines



It is worth noting that, as the number of machines increases, the common due window become more restricted. With this said, it is clear that RN-SEA-4 outperforms the other constructive algorithms. Note that the IG is more beneficial for a small number of machines. With a larger window, the stochastic local search is capable of finding better solutions since more jobs may be allocated to J_W. In this context, RN-SEA is proved to be more effective in relation to solution quality than RN-RGH.

Figure 8 – Estimated marginal means of RPD values according to window configuration

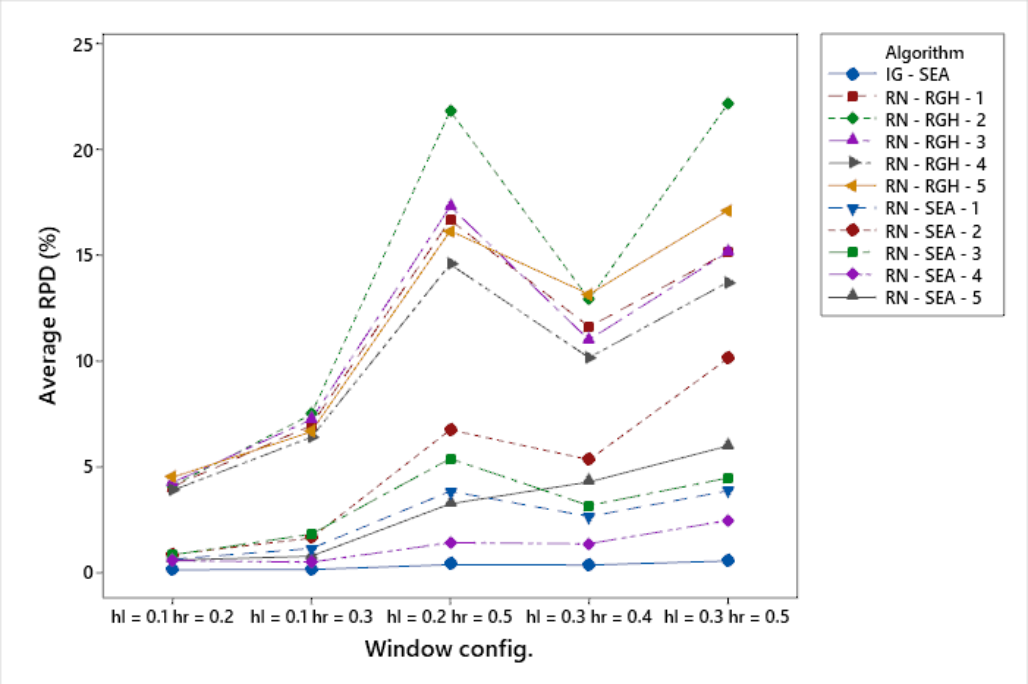


Figure 8 also demonstrates the superiority of RN-SEA against RN-RGH for problems with different window configurations. Moreover, it is possible to observe that the good performance is maintained for all cases, regardless of the position where the common due window starts and its size. From Figure 8, instances with more restrictive due windows (the ones that have a tighter left boundary factor h_l) show a smaller variation of RPD values and hence combining the IG with other heuristics may lead to smaller improvements. As the common due windows become looser, better gains are achieved, specially when their sizes are larger.

Table 21 – Test of between-subjects effects

Source	Type III sum of squares	df	Mean square	F	Sig.
Corrected model	46.59	824	0.06	26.33	0.00
Intercept	35.29	1	35.29	16435.05	0.00
n (A)	4.99	4	1.25	580.41	0.00
Algorithm (B)	17.62	10	1.76	820.45	0.00
Window (C)	8.20	4	2.05	654.68	0.00
m (D)	0.24	2	0.12	54.85	0.00
AB	4.00	40	0.10	46.59	0.00
AC	1.96	16	0.12	56.95	0.00
AD	0.39	8	0.48	22.42	0.00
BC	4.05	40	0.10	47.12	0.00
BD	0.58	20	0.03	13.46	0.00
CD	0.14	8	0.17	7.98	0.00
ABC	2.43	160	0.02	7.07	0.00
ABD	0.83	80	0.10	4.84	0.00
ACD	0.21	32	0.01	3.01	0.00
BCD	0.37	80	0.01	2.12	0.00
ABCD	0.61	320	0.00	0.89	0.93
Error	15.94	7425	0.00		
Total	97.82	8250			
Corrected total	62.53	8249			

To understand the differences between the means shown in Figures 6, 7, and 8, it was conducted an analysis of variance (ANOVA) using IBM® SPSS(™). ANOVA allows to interpret the effects between groups that have been split on two or more independent variables on the dependent variable. In this case, the independent variables are the number of jobs n (A), the *Algorithm* (B), the *Window* configuration (C), and the number of machines m (D). The dependent variable is given as the solution quality, which is measured in terms of the RPD. The results of the ANOVA experiment are illustrated in Table 21. Note that the column source represents the independent variable and their interactions, i.e., row BD of the column source shows the interactions between *Algorithm* and m. The other columns give the type III sum of squares, degrees of freedom (df), mean square, F values, and the significance values. For precise definitions of this nomenclature,

refer to Field (2018).

The experiment was set to have a significance level $\alpha = 5\%$ and two hypothesis were tested:

- **H0**: the means of RPD are the same across the methods. In other words, there are no significant statistical differences between the means of the distributions of the RPD.
- **H1**: The means of the distributions of the RPD are different.

From Table 21 it is possible to conclude that the interactions where the significance value is less than 0.05 indicate that at least one mean of the distributions of the RPD is different, that is, the null hypothesis can be rejected. It is worth noting that all of the interactions have different means, except ABCD. Consequently, this fact supports the idea that there are different means of RPD for the eleven algorithms. Moreover, it seems that the instance size affects the performance of the algorithms (row AB) together with the number of machines (row BD), and the window size (row BC).

Hitherto, it was identified the effects of different factors on the dependent variable RPD. The ANOVA experiment showed that there are statistically significant differences between means from various groups, however, it was not possible to detect in which elements from the groups these differences occur. This fact corroborates with the idea of applying a *post-hoc* analysis. Here, this analysis is conducted by means of Tukey's honestly significant difference (HSD) test.

Table 22 – Homogeneous subsets

	Subset for $\alpha = 5\%$						
Algorithm	1	2	3	4	5	6	7
IG_{SEA-4}	0.00						
RN-SEA-4	0.01	0.01					
RN-SEA-1		0.02	0.02				
RN-SEA-5			0.03				
RN-SEA-3			0.03				
RN-SEA-2				0.05			
RN-RGH-4					0.10		
RN-RGH-1					0.11	0.11	
RN-RGH-3						0.11	
RN-RGH-5						0.12	
RN-RGH-2							0.14
Sig.	0.36	0.07	0.78	1.00	0.09	0.86	1.00

Table 22 shows the homogeneous subsets for the algorithms in relation to the dependent variable RPD for a sample size of 750, which relates to the best solution found by each algorithm for all considered instances. Note that two hypothesis being analysed by

Tukey's HSD are the same for the ANOVA experiment. Here, for the sake of simplicity, the algorithms are ordered in increasing order of the values of the means. For instance, $IG_{SEA\text{-}4}$ have a smaller average RPD value in comparison with RN-SEA-4, RN-SEA-1, and so on.

With that said, it is evidenced that Tukey's HSD found 7 different homogeneous subsets, that is, groups of algorithms that share similar values of average RPD. The best performance is clearly achieved by $IG_{SEA\text{-}4}$, however, there is no statistically significant difference between the average RPD values found by RN-SEA-4. Observe that the last row of the table indicates the significance value for each subset. Taking subset 1 as an example, the significance value is 0.36, which is greater than 0.05 demonstrating that there is not enough evidence to reject the null hypothesis.

Table 22 also permits the interpretation that initial assignment rules similarly influence the performance of sequencing strategies. For instance, rule 4 is the best initial assignment strategy for both RN-SEA and RN-RGH. Moreover, two similar subsets are identified. Subsets 2 and 5 shows that RN-SEA-4 and RN-SEA-1 together with RN-RGH-4 and RN-RGH-1 share comparable means. An analogous behaviour occurs with subsets 3 an 6, which shows that, despite the fact that rules 1, 3, and 5 provide a relatively different performance when combined with RN-SEA and RN-RGH, there is not enough evidence indicating performance superiority of these rules in both sequencing strategies. Ultimately, it is possible to conclude that rule 2 has the worst performance of all initial assignment strategies as depicted in subsets 4 and 7.

Other important measure refers to the computing efforts applied to obtain the solutions previously discussed. The 95% confidence intervals of mean CPU times are shown in Figure 9 in relation to number of jobs (instance size). When the heuristics are compared in terms of the sequencing strategy on each machine, it is observed that the family of RN-RGH heuristics requires less computational effort in relation to RN-SEA. This type of behaviour occurs specially when the number of jobs exceeds 200, i.e., comparing two heuristics with the same initial assignment rules such as RN-RGH-3 and RN-SEA-3, it is seen that the average computing time of RN-RGH-3 for instances with 500 jobs is far inferior to the one in RN-SEA-3. Note that, the average computing time of RN-RGH-3 is between 3.5 seconds while RN-SEA-3 is around 80 seconds. Taking Figure 6 as a reference, it is also possible to see that for a instance size of 500, RN-SEA-3 presents lower RPD values in relation to RN-RGH-3. This fact indicates that the decision maker is able to choose the heuristic that best fits the production environment. Assuming a more general perspective, if time is crucial, the family of heuristics involving RN-RGH performs better. On the other hand, better solutions are expected to be found using RN-SEA. From a particular standpoint, RN-SEA-4 seems to have the best trade-off between computational effort and solution quality as evidenced in Figures 9 and 6.

It is important to mention that alternative data structures, that is, the ones with dif-

Figure 9 – 95% confidence intervals of mean CPU times in relation to instance size

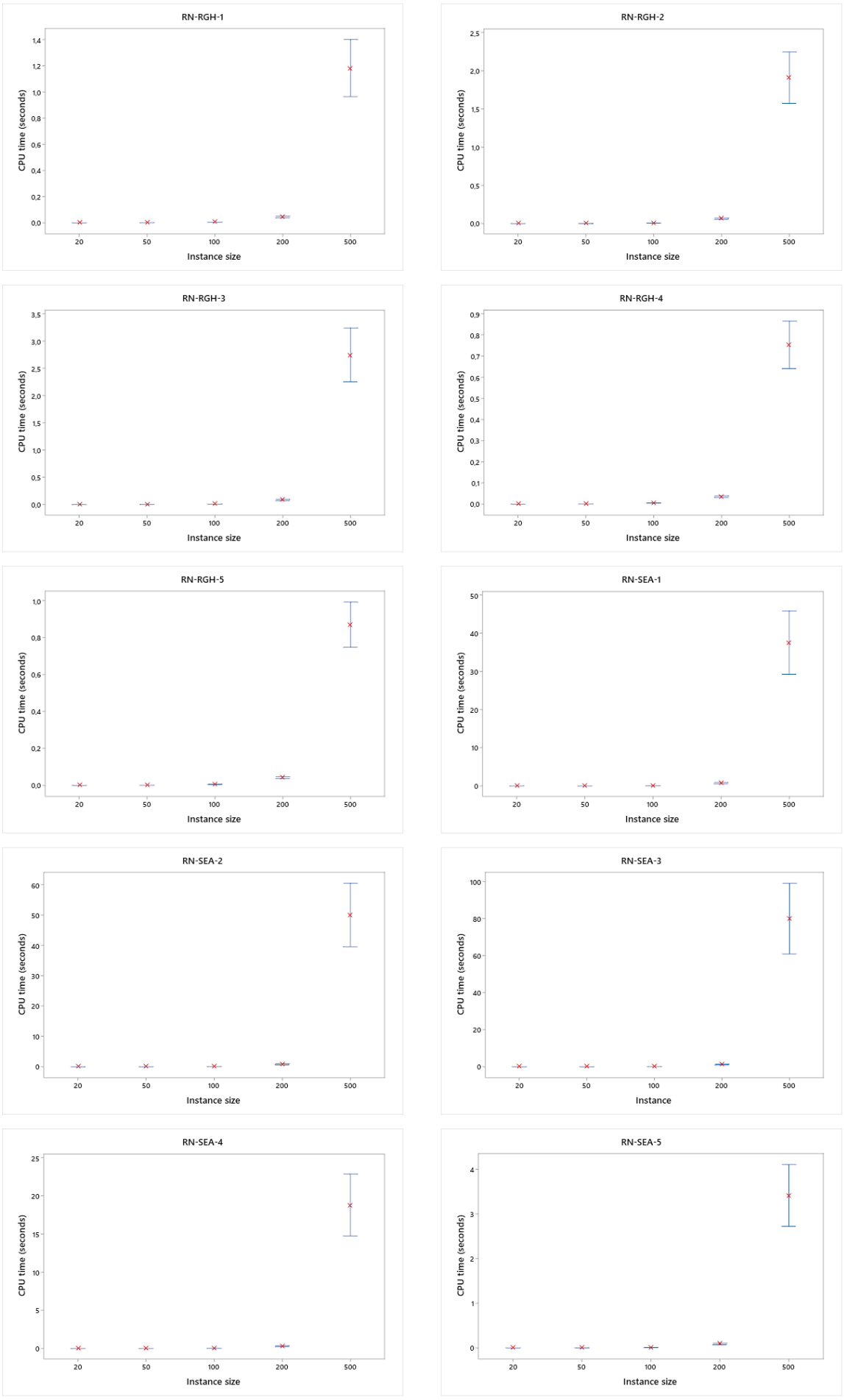


Figure 10 – 95% confidence intervals of mean CPU times in relation to the number of machines

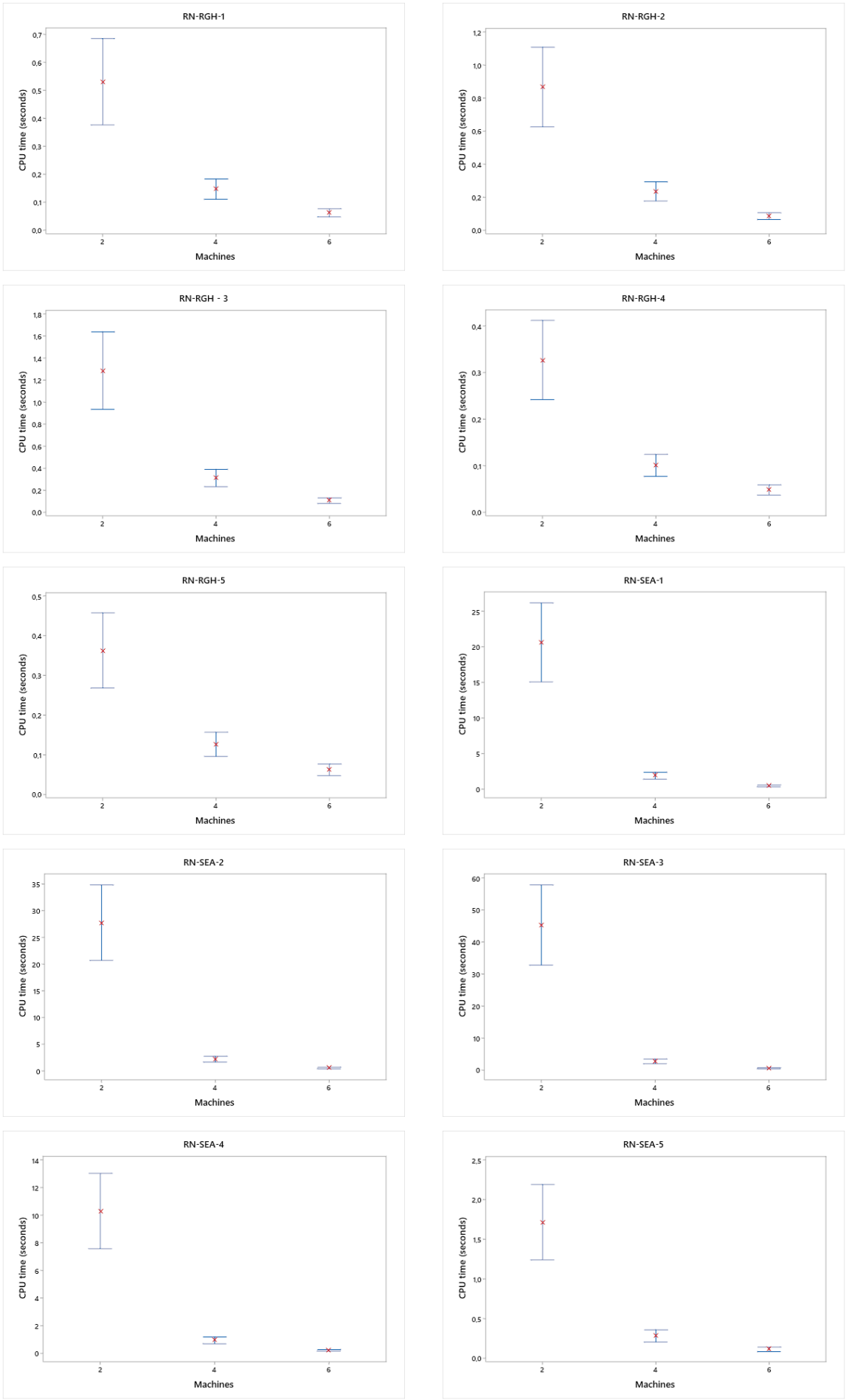
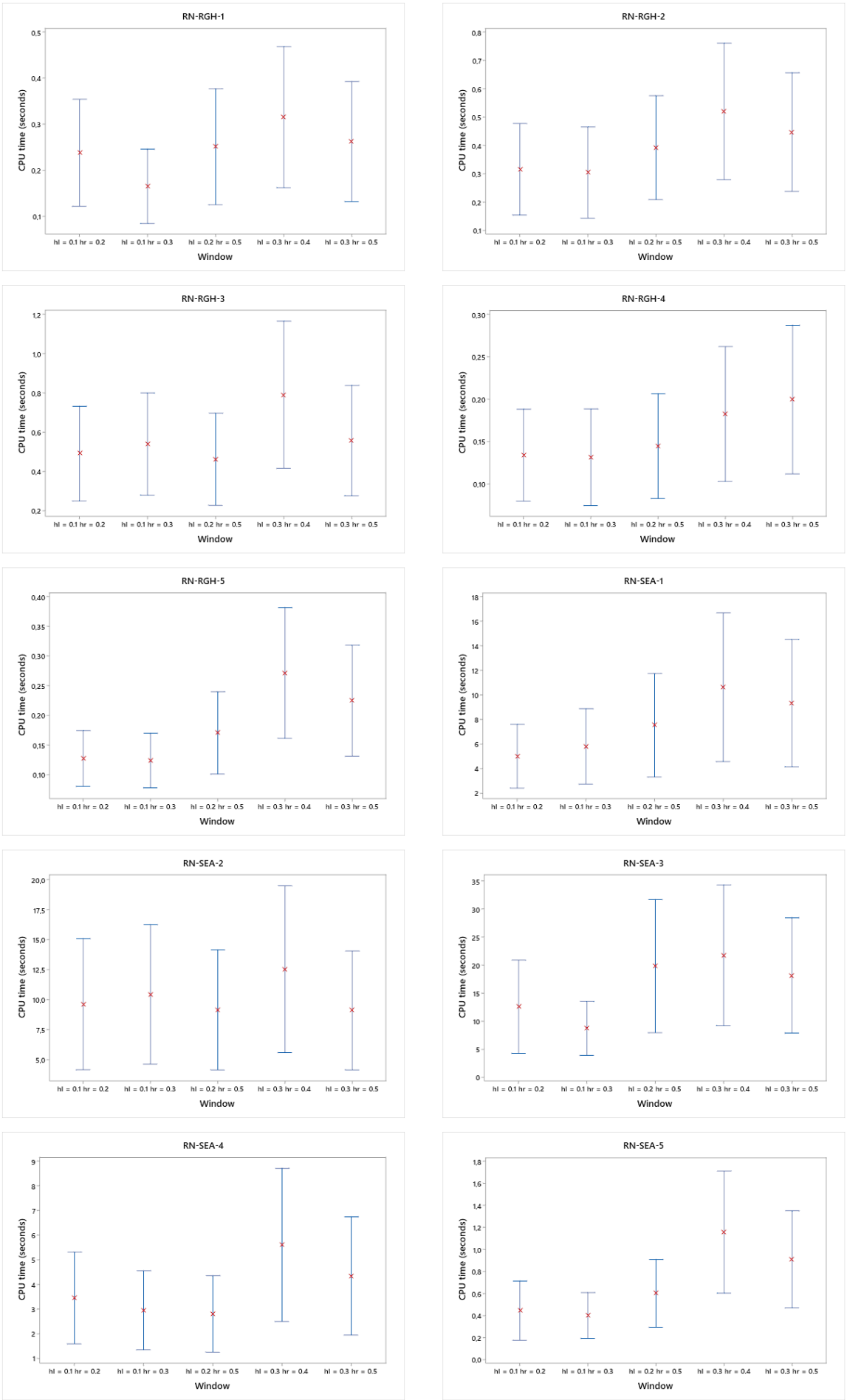


Figure 11 – 95% confidence intervals of mean CPU times in relation to the window configuration



ferent values regarding the interval in which processing times and penalties are generated, might lead to different results. It also must be stressed that the number of machines influence the average computing times of the algorithms. Figure 10 depicts 95% confidence intervals of the mean CPU times in relation to the number of machines. It is seen that the confidence intervals for CPU times follow are similar regardless of the heuristic under consideration, i.e., the CPU times for instances where $m = 2$ are usually higher in comparison to $m = 4$ and $m = 6$. As a matter of fact, as the number of machines increases, the computing efforts to solve the problem tend to decrease. Such a counter-intuitive result is also consistent with other studies in the same field (CHEN; POWELL, 1999; CHEN; LEE, 2002). The cause of this effect may relate to the smaller number of jobs assigned to each machine, which may reduce the number of combinations tested by the heuristics. In contrast with the instance size and the number of machines, the window configuration does not seem to alter the computing times. Figure 8 shows the confidence intervals of mean CPU times in relation to the window configuration. It is possible to observe that for almost all heuristics, the confidence intervals for diverse window configurations tend to overlap, which might indicate that these means are not significantly different. Again, this fact corroborates with other results previously reported in literature (CHEN; LEE, 2002).

Chapter 5

Conclusion

In this study, the state-of-the-art of ET scheduling problems involving common due dates and windows was presented. It can be noticed that scheduling problems in a just-in-time context share a strong practical importance and have been considered in the scientific literature for many years. With that said, we discussed 26 structural properties that characterize optimal solutions for single, parallel, and flow shop environments by carefully analyzing problems where the common due dates and windows are both decision variables and given parameters. In addition, we also propose a family of heuristics based on five initial ordering rules, an assignment strategy, a local search that considers the charge of earliness and tardiness penalties as well as processing times, two sequencing strategies (RN-RGH and RN-SEA) on each machine, and an IG algorithm, which is used to improve the solutions of the best performing heuristic. The rest of this chapter is structured as follows: first, we conjecture some possible trends in ET scheduling according to the literature review presented in Chapter 2; ultimately, we draw some conclusions on the results of the proposed methodology and discuss some viable enhancements that might be the topic of a future research.

5.1 Recent research and trends

This section is devoted to present the papers of a specific niche of literature that do not fit in the standards and classification of previous sections. Evidently, these types of approaches are recent (published in the last ten years) and could be extended to other machine environments and categories, i.e., inclusion of set-up times, generalization of the objective function, diversification of due date assumptions, etc. Besides, we also discuss some papers with batching in parallel machines due to the current limitations in this field.

Yin et al. (2013) show different single-machine problems involving a common due date assignment, past-sequence-dependent (p-s-d) delivery times, and position-dependent learning effects. The p-s-d delivery time is needed to remove any waiting time-induced adverse effects on job's condition (KOULAMAS; KYPARISIS, 2010). Such effect can be

computed as $q_{[j]} = \eta W_{[j]}$, where q is the p-s-d delivery time, $\eta \geq 0$ is a normalizing constant, and $W_{[j]}$ denotes the waiting time of the job in position $[j]$ in sequence. The authors show exact polynomial time procedures that require at most $O(n^4)$ time.

Low et al. (2015) introduce the problem where a single machine is subject to an availability period, that is, the machine stops processing jobs due to a planned maintenance operation. The authors show a mathematical model, a matching algorithm for the special case when $t_s \leq d \leq t_e$ (t_s and t_e denote the starting time and ending time of unavailability period), and an ant colony optimization procedure for larger instances. Later, Low, Li and Wu (2016) propose several mathematical formulations and a dynamic programming algorithm, which is suitable for finding optimal solutions for larger instances in comparison with the mathematical models.

Xiong et al. (2017) address the single-machine case with potential machine disruption. This particularity implies that a disruption will happen at a particular time that is known in advance and the disruption will last for a certain duration with a certain probability. If a job is disrupted during its processing, and it does not need to restart after the machine becomes available again, it is said to be resumable. On the other hand, if it needs to restart, it is said to be non-resumable. The authors approach both cases and develop algorithms based on dynamic programming. Furthermore, they show that the non-resumable case is NP-hard in the ordinary sense. For the resumable case, whether it is NP-hard is still an open question.

Further insights on structural properties of the problem involving machine disruptions are given by Bulbul, Kedad-Sidhoum and Şen (2018), who consider several variants of the problem for a MAD objective function and an unrestricted common due date. These variants are differentiated by the number of breaks, the job resumability scheme, and the position of the due date in relation to the break if there is only one break in the scheduling horizon. Note that the authors also cover semi-resumable jobs, which consist of jobs that are interrupted, but may be resumed at the expense of some extra processing time.

Zhu et al. (2015) consider the single-machine problem with position-dependent processing times with a rate-modifying activity, which is both deteriorating and compressible. Furthermore, both time-dependent and position-dependent models regarding the duration of the rate-modifying activity are examined. It is worth mentioning that it is also possible to allocate an amount of resource $x \in [0, x_{\max}]$, where $x_{\max}$ is the maximum extra resource level that can be used in processing the rate-modifying activity. Moreover, the compression cost resulting from the additional application of x units of resource is found by a given continuous increasing function $g(x)$. The objective is to find a schedule, the position and size of the due window, and whether or when to perform the rate-modifying activity and the amount of resource allocated to it. For the time-dependent and position-dependent compressible maintenances, the authors show $O(n^4)$ solution procedures.

Zhang et al. (2016) show a due-window assignment problem with the unusual assump-

tion of linear non-increasing processing times. The actual processing time of a job p_j is given as $p_j = \hat{p}_j(1-\omega S_j)$, where $\hat{p}_j$ is the normal processing time and $\omega > 0$ is a constant. The authors show that the optimal solution can be achieved in $O(n \log n)$. It is important to notice that this function can be seen as a variation of the learning model introduced in Biskup (1999) since it assumes a non-increasing function to determine processing times.

Table 23 – Summary of recent ET scheduling algorithms

Problem	Complexity	Reference	Algorithm
$1\|\langle d\rangle, lrn, q_{\text{p-s-d}}\|\sum(\alpha E_j + \beta T_j + \gamma d)$	P	Yin et al. (2013)	$O(n \log n)$
$1\|\langle d_l, d_r\rangle\|$ $\sum w_{[k]}(E_{[k]} + T_{[k]}) + w_0 d_l + w_{n+1}(d_r - d_l)$	P	Wang et al. (2020)	$O(n \log n)$
			$O(n^2)$
$1\|\langle d_l, d_r\rangle, lrn\|$ $\sum(\alpha E_j + \beta T_j + \gamma d_l + \delta(d_r - d_l))$	P	Zhang et al. (2016)	$O(n \log n)$
$1\|\langle d_l, d_r\rangle, dtr(rm)\|$ $\sum(\alpha E_j + \beta T_j + \gamma d_l + \delta(d_r - d_l)) + g(x)$	P	Zhu et al. (2015)	$O(n^4)$
$1\|\langle \hat{d}\rangle, avlb\|\sum(E_j + T_j)$	NP-hard	Low et al. (2015)	
			Heuristic
		Low, Li and Wu (2016)	*Enumer.*
			DP
$1\|\langle d\rangle, avlb, res, nonres\|\sum(\alpha E_j + \beta T_j + \gamma d)$	NP-hard	Xiong et al. (2017)	*DP*
$1\|\langle d\rangle, avlb, res, nonres, semires\|$ $\sum(E_j + T_j)$	NP-hard	Bulbul, Kedad-Sidhoum and Şen (2018)	*DP*
			$O(n)$
			$O(n^2)$
$1\|\langle d\rangle, \mathbf{B}, f\|\sum(E_j + T_j)$	NP-hard	Rocholl and Mönch (2018)	*Enumer.*
			Heuristic
$P\|\langle d\rangle, \mathbf{B}, f\|\sum(E_j + T_j)$	NP-hard	Rocholl and Mönch (2019)	*Enumer.*
			Heuristic
$P\|\langle d\rangle, \mathbf{B}\|\sum(E_j + T_j)$	NP-hard	Mönch and Unbehaun (2007)	*Enumer.*
			Heuristic
$P\|\langle d\rangle, f, ST_{si,b}\|\sum(\alpha_j E_j + \beta_j T_j)$	NP-hard	Yi and Wang (2003)	*Enumer.*
			Heuristic

Recently, Wang et al. (2020) addressed an unusual problem with position-dependent weights, that is, if job j is scheduled in position $[k]$, it is assigned a weight $w_{[k]}$. In other words, the weight does not correspond with the job, but with the position in which this

job is scheduled. They show that the problem with common due window assignment and position-dependent weights is polynomially solvable in $O(n \log n)$. If job-dependent learning effects are incorporated into the problem (MOSHEIOV; SIDNEY, 2003), a solution is possible in $O(n^3)$.

Rocholl and Mönch (2018) present a batch single-machine problem that frequently appear on the semiconductor industry. The objective is to form batches from multiple orders that belong to incompatible families and find a sequence that minimizes the MAD objective function. Orders that belong to the same family have the same processing time, but may have different sizes. Furthermore, the machine has a fixed capacity that cannot be exceeded and all orders that compose a batch j has the same completion time C_j. The authors show that this problem is NP-hard even when all jobs belong to the same family. They propose a mathematical model, constructive heuristics, and genetic algorithms. Rocholl and Mönch (2019) extend these results to the case of multiple parallel machines and show similar solution procedures.

It is worth mentioning that ET scheduling with batching and common due dates (windows) in parallel machine environments remains limited to few papers. To the best of our knowledge, two other approaches can be found in current literature. Yi and Wang (2003) studied the problem of scheduling job batches (each batch j may have asymmetric ET weights) about a common due date with sequence independent set-up times. The authors show a genetic algorithm with fuzzy logic operators and a mathematical model. Mönch and Unbehaun (2007) extended the problem reported in Mönch, Unbehaun and Choung (2006) (see subsection 2.2.5) to the parallel machine case without the maximum allowable tardiness constraint and, again, proposed a mathematical model, constructive heuristics as well as genetic algorithms.

Table 23 summarizes recent research and possible trends in scheduling literature. This table presents some particular notations, where $q_{\text{p-s-d}}$ stands for past-sequence-dependent delivery times, *avlb* is related to problems with machine availability constraints, and *res*, *nonres*, and *semires* are associated with the models that assume resumable, non-resumable, and semi-resumable job processing, respectively.

As seen in the summary tables throughout the text, many problems for single and parallel machines have a polynomial time solution as long as the common due dates (or windows) remain unrestrictedly large. On the other hand, if they are restricted, efficient solution procedures are not expected to be found even for MAD minimization in a single machine. The summary given in the tables allows the association of current ET literature with different objective functions, their complexity, special characteristics, and the respective machine environment. It is worth noting that the number of papers dealing with flow shops is limited in comparison with single and parallel machines. In this type of environment, a special attention was given to two-machine problems. In terms of open or job shops, literature seems scarce, however, Lauff and Werner (2004a) presented

some interesting results on the complexity of these problems with ET criteria subject to common due dates. With that said, it would be interesting to see research on these topics, specially the development of exact and heuristic methods.

Within the past few years, noticeable trends consist of combining pure ET scheduling problems with other phenomena such as resource allocation, deterioration, learning, machine disruption, maintenance, or even a combination of these effects. We strongly believe that they are emerging research trends from a theoretical or mathematical point of view, however, it is still difficult to find reasonable practical motivations in current literature.

In our opinion, there could be more efforts dedicated towards the analysis of problems in which there are multiple common due dates or multiple common due windows. To the best of our knowledge, the second case has never been addressed before. The inclusion of set-up times is still limited under the perspective of ET scheduling. In addition, some research regarding the application of common due date assignment and scheduling could be extend to open and job shops. Since we believe that practical motivations will have a significant impact on future trends, these scenarios seem more applicable to a larger number of manufacturing environments.

5.2 Methodological remarks and future directions

Besides the extensive literature survey, we have studied the identical parallel machine problem $P|\langle \hat{d}_l, \hat{d}_r \rangle| \sum (\alpha_j E_j + \beta_j T_j)$ according to the notation established in Chapter 2. The goal is to minimize the total weighted earliness and tardiness penalties when all the jobs share a common restrictive due window. Several algorithms were designed according to the properties of the problem under consideration. The results were evaluated between the heuristics due to the fact that there is no other similar approach for the same problem. As a matter of fact, only Chen and Lee (2002) proposed some exact solution procedures, however, they report that only instances with up to 40 jobs in size can be solved in reasonable time. Actually, their algorithms are quite complex and require a lot of implementation efforts from a computational perspective, therefore, our objective was to develop easy implementation algorithms capable of solving instances with up to 500 jobs in size. Our computational studies confirm that the most time consuming heuristic have an average CPU time of approximately 80 seconds, which confirms the possibility of application in practical contexts. Furthermore, analyzing the whole set of heuristics and their results allows to notice many aspects in the behavior of the algorithms:

- If the number of jobs is taken into consideration, it is possible to see clear differences in the performance of the heuristics. For all instance sizes, $\text{IG}_{\text{SEA-4}}$ was the best performing method. This is trivial due to the fact that it is the only algorithm combined with the stochastic local search procedure. In relation to the constructive heuristics, the best performance was achieved by RN-SEA-4. The reason for such

effect may rely on the fact that the structure of the data favors the fourth assignment rule. Note that RN-RGH-4 is also the best performing heuristic regarding the number of jobs if it is taken into consideration heuristics with a similar sequencing principle on each machine, that is, the family of heuristics defined by RN-RGH. It is worth noting that the earliness and tardiness penalties are generated independently of each other and of the processing times. Thus, this produces all kinds of combinations in the characteristics of the jobs. Of course, changing the way in which the instances are generated may influence the performance of the algorithms.

- ❏ Looking at different phases of the algorithms, one can see that there is a clear difference in performance between RN-SEA and RN-RGH families. RN-SEA produces the best solutions, in many cases, independently of the initial ordering rule. In contrast, it also consumes more time to find the solutions in relation to RN-RGH. Note that the initial assignment rules and the local search is is the same for both families. This behavior occurs due to the fact that RN-RGH might lead to poor results if there are many jobs with $\beta_j \gg \alpha_j$, where α_j is near to one and d_l is relatively small.

- ❏ The stochastic local search procedure has been designed as an alternative way for combining solutions in previous phases of the heuristics. One advantage of our method is related to the possibility of combining several algorithms to produce good quality solutions and concomitantly preserving reasonable computing times. For our data set, RN-SEA-4 produced the best results, however, the practitioners may refer to the other algorithms to test the best heuristic for their data set.

- ❏ The computing times are highly dependent on two factors: the size of the instances and the number of machines. As seen in the previous chapter, the window configuration does not seem to interfere that much in relation to the computing efforts of the heuristics. Again, the cause of this effect may relate to the smaller number of jobs assigned to each machine, which may reduce the combinations tested by the algorithms.

One line of future research, would be to take advantage of designing parallel or cooperative algorithms, that is, several heuristics can be run by different cores of the processor at the same time for observing the results and taking the best one. Moreover, parallel genetic algorithms and other metahueristics could be tested. Parallel computing has the potential of benefiting the process of finding good solutions to many combinatorial optimization problems. Another line of research would be the extension of this study to other related problems, such as the case of jobs with different due windows, sequence-dependent setup times, or batching.

Bibliography

ADAMOPOULOS, G. I.; PAPPIS, C. P. Scheduling under a common due-data on parallel unrelated machines. **European Journal of Operational Research**, 1998. Elsevier BV, v. 105, n. 3, p. 494–501, mar 1998.

AHMED, M. U.; SUNDARARAGHAVAN, P. S. Minimizing the weighted sum of late and early completion penalties in a single machine. **IIE Transactions**, 1990. Informa UK Limited, v. 22, n. 3, p. 288–290, sep 1990.

AKKER, M. V. den; HOOGEVEEN, H.; VELDE, S. V. de. Combining column generation and lagrangean relaxation to solve a single-machine common due date problem. **INFORMS Journal on Computing**, 2002. Institute for Operations Research and the Management Sciences (INFORMS), v. 14, n. 1, p. 37–51, feb 2002.

ALIDAEE, B.; AHMADIAN, A. Two parallel machine sequencing problems involving controllable job processing times. **European Journal of Operational Research**, 1993. Elsevier BV, v. 70, n. 3, p. 335–341, nov 1993.

ALIDAEE, B.; DRAGAN, I. A note on minimizing the weighted sum of tardy and early completion penalties in a single machine: A case of small common due date. **European Journal of Operational Research**, 1997. Elsevier BV, v. 96, n. 3, p. 559–563, feb 1997.

ALIDAEE, B.; PANWALKAR, S. S. Single stage minimum absolute lateness problem with a common due date on non-identical machines. **Journal of the Operational Research Society**, 1993. JSTOR, v. 44, n. 1, p. 29, jan 1993.

ALIDAEE, B.; WOMER, N. K. Scheduling with time dependent processing times: Review and extensions. **Journal of the Operational Research Society**, 1999. Informa UK Limited, v. 50, n. 7, p. 711–720, jul 1999.

ALLAHVERDI, A. et al. A survey of scheduling problems with setup times or costs. **European Journal of Operational Research**, 2008. Elsevier BV, v. 187, n. 3, p. 985–1032, jun 2008.

ALVAREZ-VALDES, R. et al. Minimizing weighted earliness–tardiness on a single machine with a common due date using quadratic models. **TOP**, 2012. Springer Nature, v. 20, n. 3, p. 754–767, nov 2012.

ALVAREZ-VALDES, R.; TAMARIT, J.; VILLA, F. Minimizing weighted earliness–tardiness on parallel machines using hybrid metaheuristics. **Computers & Operations Research**, 2015. Elsevier BV, v. 54, p. 1–11, feb 2015.

ARROYO, J. E. C.; LEUNG, J. Y.-T. An effective iterated greedy algorithm for scheduling unrelated parallel batch machines with non-identical capacities and unequal ready times. **Computers & Industrial Engineering**, 2017. Elsevier BV, v. 105, p. 84–100, mar 2017.

AZIZOGLU, M.; WEBSTER, S. Scheduling about an unrestricted common due window with arbitrary earliness/tardiness penalty rates. **IIE Transactions**, 1997. Informa UK Limited, v. 29, n. 11, p. 1001–1006, nov 1997.

AZIZOGLU, M.; WEBSTER, S. Scheduling job families about an unrestricted common due date on a single machine. **International Journal of Production Research**, 1997. Informa UK Limited, v. 35, n. 5, p. 1321–1330, may 1997.

BAGCHI, U.; CHANG, Y. L.; SULLIVAN, R. S. Minimizing absolute and squared deviations of completion times with different earliness and tardiness penalties and a common due date. **Naval Research Logistics**, 1987. Wiley, v. 34, n. 5, p. 739–751, oct 1987.

BAGCHI, U.; SULLIVAN, R. S.; CHANG, Y. L. Minimizing mean absolute deviation of completion times about a common due date. **Naval Research Logistics**, 1986. Wiley, v. 33, n. 2, p. 227–240, may 1986.

BAGCHI, U.; SULLIVAN, R. S.; CHANG, Y.-L. Minimizing mean squared deviation of completion times about a common due date. **Management Science**, 1987. Institute for Operations Research and the Management Sciences (INFORMS), v. 33, n. 7, p. 894–906, jul 1987.

BAKER, K. R.; SCUDDER, G. D. On the assignment of optimal due dates. **Journal of the Operational Research Society**, 1989. JSTOR, v. 40, n. 1, p. 93, jan 1989.

BAKER, K. R.; SCUDDER, G. D. Sequencing with earliness and tardiness penalties: A review. **Operations Research**, 1990. Institute for Operations Research and the Management Sciences (INFORMS), v. 38, n. 1, p. 22–36, feb 1990.

BANK, J.; WERNER, F. Heuristic algorithms for unrelated parallel machine scheduling with a common due date, release dates, and linear earliness and tardiness penalties. **Mathematical and Computer Modelling**, 2001. Elsevier BV, v. 33, n. 4-5, p. 363–383, feb 2001.

BEASLEY, J. E. OR-library: Distributing test problems by electronic mail. **Journal of the Operational Research Society**, 1990. Informa UK Limited, v. 41, n. 11, p. 1069–1072, nov 1990.

BECTOR, C. R.; GUPTA, Y. P.; GUPTA, M. C. Determination of an optimal common due date and optimal sequence in a single machine job shop. **International Journal of Production Research**, 1988. Informa UK Limited, v. 26, n. 4, p. 613–628, apr 1988.

BEYRANVAND, M. S.; PEYGHAMI, M. R.; GHATEE, M. On the quadratic model for unrelated parallel machine scheduling problem with restrictive common due date. **Optimization Letters**, 2012. Springer Nature, v. 6, n. 8, p. 1897–1911, sep 2012.

BIRGIN, E. G.; FERREIRA, J.; RONCONI, D. A filtered beam search method for the m-machine permutation flowshop scheduling problem minimizing the earliness and tardiness penalties and the waiting time of the jobs. **Computers & Operations Research**, 2020. Elsevier BV, v. 114, p. 104824, feb 2020.

BIRGIN, E. G.; RONCONI, D. P. Heuristic methods for the single machine scheduling problem with different ready times and a common due date. **Engineering Optimization**, 2012. Informa UK Limited, v. 44, n. 10, p. 1197–1208, oct 2012.

BISKUP, D. Single-machine scheduling with learning considerations. **European Journal of Operational Research**, 1999. Elsevier BV, v. 115, n. 1, p. 173–178, may 1999.

BISKUP, D.; CHENG, T. C. E. Multiple-machine scheduling with earliness, tardiness and completion time penalties. **Computers & Operations Research**, 1999. Elsevier BV, v. 26, n. 1, p. 45–57, jan 1999.

BISKUP, D.; CHENG, T. C. E. Single machine scheduling with controllable processing times and earliness, tardiness and completion times penalties. **Engineering Optimization**, 1999. Informa UK Limited, v. 31, n. 3, p. 329–336, feb 1999.

BISKUP, D.; FELDMANN, M. Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. **Computers & Operations Research**, 2001. Elsevier BV, v. 28, n. 8, p. 787–801, jul 2001.

BISKUP, D.; FELDMANN, M. On scheduling around large restrictive common due windows. **European Journal of Operational Research**, 2005. Elsevier BV, v. 162, n. 3, p. 740–761, may 2005.

BISKUP, D.; JAHNKE, H. Common due date assignment for scheduling on a single machine with jointly reducible processing times. **International Journal of Production Economics**, 2001. Elsevier BV, v. 69, n. 3, p. 317–322, feb 2001.

BISKUP, D.; SIMONS, D. Common due date scheduling with autonomous and induced learning. **European Journal of Operational Research**, 2004. Elsevier BV, v. 159, n. 3, p. 606–616, dec 2004.

BULBUL, K.; KEDAD-SIDHOUM, S.; ŞEN, H. Single-machine common due date total earliness/tardiness scheduling with machine unavailability. **Journal of Scheduling**, 2018. Springer Science and Business Media LLC, v. 22, n. 5, p. 543–565, sep 2018.

CHANDRA, P.; MEHTA, P.; TIRUPATI, D. Permutation flow shop scheduling with earliness and tardiness penalties. **International Journal of Production Research**, 2009. Informa UK Limited, v. 47, n. 20, p. 5591–5610, jul 2009.

CHEN, W.-Y.; SHEEN, G.-J. Single-machine scheduling with multiple performance measures: Minimizing job-dependent earliness and tardiness subject to the number of tardy jobs. **International Journal of Production Economics**, 2007. Elsevier BV, v. 109, n. 1-2, p. 214–229, sep 2007.

CHEN, Z.-L. Scheduling and common due date assignment with earliness-tardiness penalties and batch delivery costs. **European Journal of Operational Research**, 1996. Elsevier BV, v. 93, n. 1, p. 49–60, aug 1996.

CHEN, Z.-L. Scheduling with batch setup times and earliness-tardiness penalties. **European Journal of Operational Research**, 1997. Elsevier BV, v. 96, n. 3, p. 518–537, feb 1997.

CHEN, Z. L.; LEE, C. Y. Parallel machine scheduling with a common due window. **European Journal of Operational Research**, 2002. Elsevier BV, v. 136, n. 3, p. 512–527, feb 2002.

CHEN, Z.-L.; POWELL, W. B. A column generation based decomposition algorithm for a parallel machine just-in-time scheduling problem. **European Journal of Operational Research**, 1999. Elsevier BV, v. 116, n. 1, p. 220–232, jul 1999.

CHENG, B.; CHENG, L. Note on "single-machine due-window assignment and scheduling with resource allocation, aging effect, and a deteriorating rate-modifying activity". **Computers & Industrial Engineering**, 2014. Elsevier BV, v. 78, p. 320–322, dec 2014.

CHENG, T. C. E. A duality approach to optimal due date determination. **Engineering Optimization**, 1985. Informa UK Limited, v. 9, n. 2, p. 127–130, jan 1985.

CHENG, T. C. E. An algorithm for the con due-date determination and sequencing problem. **Computers & Operations Research**, 1987. Elsevier BV, v. 14, n. 6, p. 537–542, jan 1987.

CHENG, T. C. E. An alternative proof of optimality for the common due-date assignment problem. **European Journal of Operational Research**, 1988. Elsevier BV, v. 37, n. 2, p. 250–253, nov 1988.

CHENG, T. C. E. Optimal common due-date with limited completion time deviation. **Computers & Operations Research**, 1988. Elsevier BV, v. 15, n. 2, p. 91–96, jan 1988.

CHENG, T. C. E. A heuristic for common due-date assignment and job scheduling on parallel machines. **Journal of the Operational Research Society**, 1989. Informa UK Limited, v. 40, n. 12, p. 1129–1135, dec 1989.

CHENG, T. C. E. A note on a partial search algorithm for the single-machine optimal common due-date assignment and sequencing problem. **Computers & Operations Research**, 1990. Elsevier BV, v. 17, n. 3, p. 321–324, jan 1990.

CHENG, T. C. E.; CHEN, Z.-L. Parallel-machine scheduling problems with earliness and tardiness penalties. **Journal of the Operational Research Society**, 1994. JSTOR, v. 45, n. 6, p. 685, jun 1994.

CHENG, T. C. E.; CHEN, Z. L.; SHAKHLEVICH, N. V. Common due date assignment and scheduling with ready times. **Computers & Operations Research**, 2002. Elsevier BV, v. 29, n. 14, p. 1957–1967, dec 2002.

CHENG, T. C. E.; GUPTA, M. Survey of scheduling research involving due date determination decisions. **European Journal of Operational Research**, 1989. Elsevier BV, v. 38, n. 2, p. 156–166, jan 1989.

CHENG, T. C. E.; KANG, L.; NG, C. T. Due-date assignment and single machine scheduling with deteriorating jobs. **Journal of the Operational Research Society**, 2004. Informa UK Limited, v. 55, n. 2, p. 198–203, feb 2004.

CHENG, T. C. E.; KANG, L. Y.; NG, C. T. Due-date assignment and parallel-machine scheduling with deteriorating jobs. **Journal of the Operational Research Society**, 2007. Informa UK Limited, v. 58, n. 8, p. 1103–1108, aug 2007.

CHENG, T. C. E.; OGUZ, C.; QI, X. D. Due-date assignment and single machine scheduling with compressible processing times. **International Journal of Production Economics**, 1996. Elsevier BV, v. 43, n. 2-3, p. 107–113, jun 1996.

CHENG, T. C. E.; YANG, S.-J.; YANG, D.-L. Common due-window assignment and scheduling of linear time-dependent deteriorating jobs and a deteriorating maintenance activity. **International Journal of Production Economics**, 2012. Elsevier BV, v. 135, n. 1, p. 154–161, jan 2012.

DE, P.; GHOSH, J. B.; WELLS, C. E. A note on the minimization of mean squared deviation of completion times about a common due date. **Management Sciences**, 1989. Institute for Operations Research and the Management Sciences (INFORMS), v. 35, n. 9, p. 1143–1147, sep 1989.

DE, P.; GHOSH, J. B.; WELLS, C. E. CON due-date determination and sequencing. **Computers & Operations Research**, 1990. Elsevier BV, v. 17, n. 4, p. 333–342, jan 1990.

DE, P.; GHOSH, J. B.; WELLS, C. E. Scheduling about a common due date with earliness and tardiness penalties. **Computers & Operations Research**, 1990. Elsevier BV, v. 17, n. 2, p. 231–241, jan 1990.

DE, P.; GHOSH, J. B.; WELLS, C. E. Optimal delivery time quotation and order sequencing. **Decision Sciences**, 1991. Wiley, v. 22, n. 2, p. 379–390, mar 1991.

DE, P.; GHOSH, J. B.; WELLS, C. E. On the general solution for a class of early/tardy problems. **Computers & Operations Research**, 1993. Elsevier BV, v. 20, n. 2, p. 141–149, feb 1993.

DE, P.; GHOSH, J. B.; WELLS, C. E. Due-date assignment and early/tardy scheduling on identical parallel machines. **Naval Research Logistics**, 1994. Wiley, v. 41, n. 1, p. 17–32, feb 1994.

DE, P.; GHOSH, J. B.; WELLS, C. E. Solving a generalized model for CON due date assignment and sequencing. **International Journal of Production Economics**, 1994. Elsevier BV, v. 34, n. 2, p. 179–185, mar 1994.

DIAMOND, J. E.; CHENG, T. C. E. Error bound for common due date assignment and job scheduling on parallel machines. **IIE Transactions**, 2000. Informa UK Limited, v. 32, n. 5, p. 445–448, may 2000.

DICKMAN, B.; WILAMOWSKY, Y.; EPSTEIN, S. Optimal common due-date with limited completion time. **Computers & Operations Research**, 1991. Elsevier BV, v. 18, n. 1, p. 125–127, jan 1991.

DICKMAN, B.; WILAMOWSKY, Y.; EPSTEIN, S. Multiple common due dates. **Naval Research Logistics**, 2001. Wiley, v. 48, n. 4, p. 293–298, 2001.

DILEEPAN, P. Common due date scheduling problem with separate earliness and tardiness penalties. **Computers & Operations Research**, 1993. Elsevier BV, v. 20, n. 2, p. 179–184, feb 1993.

DROBOUCHEVITCH, I. G.; SIDNEY, J. B. Minimization of earliness, tardiness and due date penalties on uniform parallel machines with identical jobs. **Computers & Operations Research**, 2012. Elsevier BV, v. 39, n. 9, p. 1919–1926, sep 2012.

EILON, S.; CHOWDHURY, I. G. Minimising waiting time variance in the single machine problem. **Management Science**, 1977. Institute for Operations Research and the Management Sciences (INFORMS), v. 23, n. 6, p. 567–575, feb 1977.

EMMONS, H. Scheduling to a common due date on parallel uniform processors. **Naval Research Logistics**, 1987. Wiley, v. 34, n. 6, p. 803–810, 1987.

ENGIN, O.; ENGIN, B. Hybrid flow shop with multiprocessor task scheduling based on earliness and tardiness penalties. **Journal of Enterprise Information Management**, 2018. Emerald, v. 31, n. 6, p. 925–936, oct 2018.

FAN, Y.-P.; ZHAO, C.-L. Single machine scheduling with multiple common due date assignment and aging effect under a deteriorating maintenance activity consideration. **Journal of Applied Mathematics and Computing**, 2014. Springer Science and Business Media LLC, v. 46, n. 1-2, p. 51–66, nov 2014.

FELDMANN, M.; BISKUP, D. Single-machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches. **Computers & Industrial Engineering**, 2003. Elsevier BV, v. 44, n. 2, p. 307–323, feb 2003.

FIELD, A. **Discovering Statistics Using IBM SPSS**. [S.l.]: Sage Publications Ltd., 2018. ISBN 1526419521.

GENG, X.-N.; WANG, J.-B.; BAI, D. Common due date assignment scheduling for a no-wait flowshop with convex resource allocation and learning effect. **Engineering Optimization**, 2018. Informa UK Limited, v. 51, n. 8, p. 1301–1323, oct 2018.

GERSTL, E.; MOSHEIOV, G. Due-window assignment problems with unit-time jobs. **Applied Mathematics and Computation**, 2013. Elsevier BV, v. 220, p. 487–495, sep 2013.

GERSTL, E.; MOSHEIOV, G. Due-window assignment with identical jobs on parallel uniform machines. **European Journal of Operational Research**, 2013. Elsevier BV, v. 229, n. 1, p. 41–47, aug 2013.

GERSTL, E.; MOSHEIOV, G. An improved algorithm for due-window assignment on parallel identical machines with unit-time jobs. **Information Processing Letters**, 2013. Elsevier BV, v. 113, n. 19-21, p. 754–759, sep 2013.

GIANNOPOULOS, N.; NEARCHOU, A. C. Bi-criteria scheduling against restrictive common due dates using a multi-objective differential evolution algorithm. **IMA Journal of Management Mathematics**, 2018. Oxford University Press (OUP), p. dpw015, sep 2018.

GORDON, V. S.; PROTH, J.-M.; CHU, C. Due date assignment and scheduling: SLK, TWK and other due date assignment models. **Production Planning & Control**, 2002. Informa UK Limited, v. 13, n. 2, p. 117–132, jan 2002.

GORDON, V. S.; PROTH, J.-M.; CHU, C. A survey of the state-of-the-art of common due date assignment and scheduling research. **European Journal of Operational Research**, 2002. Elsevier BV, v. 139, n. 1, p. 1–25, may 2002.

GORDON, V. S.; STRUSEVICH, V.; DOLGUI, A. Scheduling with due date assignment under special conditions on job processing. **Journal of Scheduling**, 2012. Springer Nature, v. 15, n. 4, p. 447–456, jun 2012.

GRAHAM, R. L. et al. (Ed.). **Optimization and approximation in deterministic sequencing and scheduling: a survey**. [S.l.]: Elsevier BV, 1979.

GUPTA, Y. P.; BECTOR, C. R.; GUPTA, M. C. Optimal schedule on a single machine using various due date determination methods. **Computers in Industry**, 1990. Elsevier BV, v. 15, n. 3, p. 245–254, nov 1990.

HALL, N. G. Single and multiple-processor models for minimizing completion time variance. **Naval Research Logistics**, 1986. Wiley, v. 33, n. 1, p. 49–54, feb 1986.

HALL, N. G.; KUBIAK, W.; SETHI, S. P. Earliness–tardiness scheduling problems, II: Deviation of completion times about a restrictive common due date. **Operations Research**, 1991. Institute for Operations Research and the Management Sciences (INFORMS), v. 39, n. 5, p. 847–856, oct 1991.

HALL, N. G.; POSNER, M. E. Earliness-tardiness scheduling problems, I: Weighted deviation of completion times about a common due date. **Operations Research**, 1991. Institute for Operations Research and the Management Sciences (INFORMS), v. 39, n. 5, p. 836–846, oct 1991.

HAO, Q. et al. Common due-date determination and sequencing using tabu search. **Computers & Operations Research**, 1996. Elsevier BV, v. 23, n. 5, p. 409–417, may 1996.

HEPDOGAN, S. et al. A meta-RaPS for the early/tardy single machine scheduling problem. **International Journal of Production Research**, 2009. Informa UK Limited, v. 47, n. 7, p. 1717–1732, feb 2009.

HERRMANN, J. W.; LEE, C.-Y. On scheduling to minimize earliness-tardiness and batch delivery costs with a common due date. **European Journal of Operational Research**, 1993. Elsevier BV, v. 70, n. 3, p. 272–288, nov 1993.

HINO, C. M.; RONCONI, D. P.; MENDES, A. B. Minimizing earliness and tardiness penalties in a single-machine problem with a common due date. **European Journal of Operational Research**, 2005. Elsevier BV, v. 160, n. 1, p. 190–201, jan 2005.

HOESEL, S. van; WAGELMANS, A.; MOERMAN, B. Using geometric techniques to improve dynamic programming algorithms for the economic lot-sizing problem and extensions. **European Journal of Operational Research**, 1994. Elsevier BV, v. 75, n. 2, p. 312–331, jun 1994.

HOOGEVEEN, J.; VELDE, S. van de. Scheduling around a small common due date. **European Journal of Operational Research**, 1991. Elsevier BV, v. 55, n. 2, p. 237–242, nov 1991.

HOOGEVEEN, J. A.; OOSTERHOUT, H.; VELDE, S. L. van de. New lower and upper bounds for scheduling around a small common due date. **Operations Research**, 1994. Institute for Operations Research and the Management Sciences (INFORMS), v. 42, n. 1, p. 102–110, feb 1994.

HOSSEINI, N.; TAVAKKOLI-MOGHADDAM, R. Two meta-heuristics for solving a new two-machine flowshop scheduling problem with the learning effect and dynamic arrivals. **The International Journal of Advanced Manufacturing Technology**, 2013. Springer Science and Business Media LLC, v. 65, n. 5-8, p. 771–786, may 2013.

İŞLER, M. C.; TOKLU, B.; ÇELIK, V. Scheduling in a two-machine flow-shop for earliness/tardiness under learning effect. **The International Journal of Advanced Manufacturing Technology**, 2012. Springer Science and Business Media LLC, v. 61, n. 9-12, p. 1129–1137, dec 2012.

JAMES, R. Using tabu search to solve the common due date early/tardy machine scheduling problem. **Computers & Operations Research**, 1997. Elsevier BV, v. 24, n. 3, p. 199–208, mar 1997.

JANIAK, A. et al. Soft due window assignment and scheduling of unit-time jobs on parallel machines. **4OR - A Quarterly Journal of Operations Research**, 2012. Springer Nature, v. 10, n. 4, p. 347–360, may 2012.

JANIAK, A. et al. Parallel machine scheduling and common due window assignment with job independent earliness and tardiness costs. **Information Sciences**, 2013. Elsevier BV, v. 224, p. 109–117, mar 2013.

JANIAK, A. et al. A survey on scheduling problems with due windows. **European Journal of Operational Research**, 2015. Elsevier BV, v. 242, n. 2, p. 347–357, apr 2015.

JI, M. et al. Single-machine due-window assignment and scheduling with resource allocation, aging effect, and a deteriorating rate-modifying activity. **Computers & Industrial Engineering**, 2013. Elsevier BV, v. 66, n. 4, p. 952–961, dec 2013.

JÓZEFOWSKA, J. (Ed.). **Just-In-Time Scheduling: Models and Algorithms for Computer and Manufacturing Systems**. [S.l.]: Springer US, 2007.

JURISCH, B.; KUBIAK, W.; JOZEFOWSKA, J. Algorithms for minclique scheduling problems. **Discrete Applied Mathematics**, 1997. Elsevier BV, v. 72, n. 1-2, p. 115–139, jan 1997.

KAHLBACHER, H. G. Scheduling with monotonous earliness and tardiness penalties. **European Journal of Operational Research**, 1993. Elsevier BV, v. 64, n. 2, p. 258–277, jan 1993.

KANET, J. J. Minimizing the average deviation of job completion times about a common due date. **Naval Research Logistics**, 1981. Wiley, v. 28, n. 4, p. 643–651, dec 1981.

KANG, Q.; HE, H.; WEI, J. An effective iterated greedy algorithm for reliability-oriented task allocation in distributed computing systems. **Journal of Parallel and Distributed Computing**, 2013. Elsevier BV, v. 73, n. 8, p. 1106–1115, aug 2013.

KELLERER, H.; RUSTOGI, K.; STRUSEVICH, V. A. A fast FPTAS for single machine scheduling problem of minimizing total weighted earliness and tardiness about a large common due date. **Omega**, 2020. Elsevier BV, v. 90, p. 101992, jan 2020.

KELLERER, H.; STRUSEVICH, V. A. Minimizing total weighted earliness-tardiness on a single machine around a small common due date: an FPTAS using quadratic knapsack. **International Journal of Foundations of Computer Science**, 2010. World Scientific Pub Co Pte Lt, v. 21, n. 03, p. 357–383, jun 2010.

KIM, J.-G.; KIM, J.-S.; LEE, D.-H. Fast and meta-heuristics for common due-date assignment and scheduling on parallel machines. **International Journal of Production Research**, 2012. Informa UK Limited, v. 50, n. 20, p. 6040–6057, oct 2012.

KIM, J.-G.; LEE, D.-H. Algorithms for common due-date assignment and sequencing on a single machine with sequence-dependent setup times. **Journal of the Operational Research Society**, 2009. Informa UK Limited, v. 60, n. 9, p. 1264–1272, sep 2009.

KOULAMAS, C. Single-machine scheduling with time windows and earliness/tardiness penalties. **European Journal of Operational Research**, 1996. Elsevier BV, v. 91, n. 1, p. 190–202, may 1996.

KOULAMAS, C. Common due date assignment with generalized earliness/tardiness penalties. **Computers & Industrial Engineering**, 2017. Elsevier BV, v. 109, p. 79–83, jul 2017.

KOULAMAS, C.; KYPARISIS, G. J. Single-machine scheduling problems with past-sequence-dependent delivery times. **International Journal of Production Economics**, 2010. Elsevier BV, v. 126, n. 2, p. 264–266, aug 2010.

KOVALYOV, M. Y.; KUBIAK, W. A fully polynomial approximation scheme for the weighted earliness–tardiness problem. **Operations Research**, 1999. Institute for Operations Research and the Management Sciences (INFORMS), v. 47, n. 5, p. 757–761, oct 1999.

KRAMER, A.; SUBRAMANIAN, A. A unified heuristic and an annotated bibliography for a large class of earliness–tardiness scheduling problems. **Journal of Scheduling**, 2019. Springer Nature, v. 22, n. 1, p. 21–57, dec 2019.

KRAMER, F. J.; LEE, C.-Y. Common due-window scheduling. **Production and Operations Management**, 1993. Wiley, v. 2, n. 4, p. 262–275, dec 1993.

KRAMER, F. J.; LEE, C. Y. Due window scheduling for parallel machines. **Mathematical and Computer Modelling**, 1994. Elsevier BV, v. 20, n. 2, p. 69–89, jul 1994.

KUBIAK, W. Completion time variance minimization on a single machine is difficult. **Operations Research Letters**, 1993. Elsevier BV, v. 14, n. 1, p. 49–59, aug 1993.

KUO, W.-H.; YANG, D.-L. Minimizing the total completion time in a single-machine scheduling problem with a time-dependent learning effect. **European Journal of Operational Research**, 2006. Elsevier BV, v. 174, n. 2, p. 1184–1190, oct 2006.

KUO, W. H.; YANG, D. L. A note on due-date assignment and single-machine scheduling with deteriorating jobs and learning effects. **Journal of the Operational Research Society**, 2011. Informa UK Limited, v. 62, n. 1, p. 206–210, jan 2011.

LAUFF, V.; WERNER, F. On the complexity and some properties of multi-stage scheduling problems with earliness and tardiness penalties. **Computers & Operations Research**, 2004. Elsevier BV, v. 31, n. 3, p. 317–345, mar 2004.

LAUFF, V.; WERNER, F. Scheduling with common due date, earliness and tardiness penalties for multimachine problems: A survey. **Mathematical and Computer Modelling**, 2004. Elsevier BV, v. 40, n. 5-6, p. 637–655, sep 2004.

LEE, C.-Y.; DANUSAPUTRO, S. L.; LIN, C.-S. Minimizing weighted number of tardy jobs and weighted earliness-tardiness penalties about a common due date. **Computers & Operations Research**, 1991. Elsevier BV, v. 18, n. 4, p. 379–389, jan 1991.

LEE, C. Y.; KIM, S. J. Parallel genetic algorithms for the earliness-tardiness job scheduling problem with general penalty weights. **Computers & Industrial Engineering**, 1995. Elsevier BV, v. 28, n. 2, p. 231–243, apr 1995.

LEYVAND, Y.; SHABTAY, D.; STEINER, G. A unified approach for scheduling with convex resource consumption functions using positional penalties. **European Journal of Operational Research**, 2010. Elsevier BV, v. 206, n. 2, p. 301–312, oct 2010.

LI, S.; NG, C.; YUAN, J. Scheduling deteriorating jobs with CON/SLK due date assignment on a single machine. **International Journal of Production Economics**, 2011. Elsevier BV, v. 131, n. 2, p. 747–751, jun 2011.

LI, Z. et al. Earliness–tardiness minimization on scheduling a batch processing machine with non-identical job sizes. **Computers & Industrial Engineering**, 2015. Elsevier BV, v. 87, p. 590–599, sep 2015.

LI, Z. et al. Bi-objective hybrid flow shop scheduling with common due date. **Operational Research**, 2019. Springer Science and Business Media LLC, mar 2019.

LIAO, C.-J.; CHENG, C.-C. A variable neighborhood search for minimizing single machine weighted earliness and tardiness with common due date. **Computers & Industrial Engineering**, 2007. Elsevier BV, v. 52, n. 4, p. 404–413, may 2007.

LIMAN, S. D.; PANWALKAR, S. S.; THONGMEE, S. Determination of common due window location in a single machine scheduling problem. **European Journal of Operational Research**, 1996. Elsevier BV, v. 93, n. 1, p. 68–74, aug 1996.

LIMAN, S. D.; PANWALKAR, S. S.; THONGMEE, S. A single machine scheduling problem with common due window and controllable processing times. **Annals of Operations Research**, 1997. Springer Nature, v. 70, p. 145–154, 1997.

LIMAN, S. D.; PANWALKAR, S. S.; THONGMEE, S. Common due window size and location determination in a single machine scheduling problem. **Journal of the Operational Research Society**, 1998. Informa UK Limited, v. 49, n. 9, p. 1007–1010, sep 1998.

LIN, S.-W.; CHOU, S.-Y.; CHEN, S.-C. Meta-heuristic approaches for minimizing total earliness and tardiness penalties of single-machine scheduling with a common due date. **Journal of Heuristics**, 2007. Springer Nature, v. 13, n. 2, p. 151–165, dec 2007.

LIN, S.-W.; CHOU, S.-Y.; YING, K.-C. A sequential exchange approach for minimizing earliness–tardiness penalties of single-machine scheduling with a common due date. **European Journal of Operational Research**, 2007. Elsevier BV, v. 177, n. 2, p. 1294–1301, mar 2007.

LIN, S. W. et al. Minimization of maximum lateness on parallel machines with sequence-dependent setup times and job release dates. **Computers & Operations Research**, 2011. Elsevier BV, v. 38, n. 5, p. 809–815, may 2011.

LIN, S. W.; LU, C. C.; YING, K. C. Minimization of total tardiness on unrelated parallel machines with sequence- and machine-dependent setup times under due date constraints. **The International Journal of Advanced Manufacturing Technology**, 2011. Springer Science and Business Media LLC, v. 53, n. 1-4, p. 353–361, jul 2011.

LIN, S.-W. et al. Minimising total weighted earliness and tardiness penalties on identical parallel machines using a fast ruin-and-recreate algorithm. **International Journal of Production Research**, 2016. Informa UK Limited, v. 54, n. 22, p. 6879–6890, may 2016.

LIN, S.-W. et al. Single machine job sequencing with a restricted common due window. **IEEE Access**, 2019. Institute of Electrical and Electronics Engineers (IEEE), v. 7, p. 148741–148755, 2019.

LIU, J.; WANG, Y.; MIN, X. Single-machine scheduling with common due-window assignment for deteriorating jobs. **Journal of the Operational Research Society**, 2014. Informa UK Limited, v. 65, n. 2, p. 291–301, feb 2014.

LOW, C.; LI, R.-K.; WU, G.-H. Minimizing total earliness and tardiness for common due date single-machine scheduling with an unavailability interval. **Mathematical Problems in Engineering**, 2016. Hindawi Limited, v. 2016, p. 1–12, 2016.

LOW, C. et al. Minimizing the sum of absolute deviations under a common due date for a single-machine scheduling problem with availability constraints. **Journal of Industrial and Production Engineering**, 2015. Informa UK Limited, v. 32, n. 3, p. 204–217, apr 2015.

MACCARTHY, B. L.; LIU, J. Addressing the gap in scheduling research: a review of optimization and heuristic methods in production scheduling. **International Journal of Production Research**, 1993. Informa UK Limited, v. 31, n. 1, p. 59–79, jan 1993.

MERTEN, A. G.; MULLER, M. E. Variance minimization in single machine sequencing problems. **Management Science**, 1972. Institute for Operations Research and the Management Sciences (INFORMS), v. 18, n. 9, p. 518–528, may 1972.

MIN, L.; CHENG, W. Genetic algorithms for the optimal common due date assignment and the optimal scheduling policy in parallel machine earliness/tardiness scheduling problems. **Robotics and Computer-Integrated Manufacturing**, 2006. Elsevier BV, v. 22, n. 4, p. 279–287, aug 2006.

MÖNCH, L. et al. A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. **Journal of Scheduling**, 2011. Springer Science and Business Media LLC, v. 14, n. 6, p. 583–599, jan 2011.

MÖNCH, L.; UNBEHAUN, R. Decomposition heuristics for minimizing earliness–tardiness on parallel burn-in ovens with a common due date. **Computers & Operations Research**, 2007. Elsevier BV, v. 34, n. 11, p. 3380–3396, nov 2007.

MÖNCH, L.; UNBEHAUN, R.; CHOUNG, Y. I. Minimizing earliness–tardiness on a single burn-in oven with a common due date and maximum allowable tardiness constraint. **OR Spectrum**, 2006. Springer Science and Business Media LLC, v. 28, n. 2, p. 177–198, dec 2006.

MONDAL, S. A. Minimization of squared deviation of completion times about a common due date. **Computers & Operations Research**, 2002. Elsevier BV, v. 29, n. 14, p. 2073–2085, dec 2002.

MONDAL, S. A.; SEN, A. K. Single machine weighted earliness–tardiness penalty problem with a common due date. **Computers & Operations Research**, 2001. Elsevier BV, v. 28, n. 7, p. 649–669, jun 2001.

MONMA, C. L.; POTTS, C. N. On the complexity of scheduling with batch setup times. **Operations Research**, 1989. Institute for Operations Research and the Management Sciences (INFORMS), v. 37, n. 5, p. 798–804, oct 1989.

MOR, B.; MOSHEIOV, G. Scheduling a deteriorating maintenance activity and due-window assignment. **Computers & Operations Research**, 2015. Elsevier BV, v. 57, p. 33–40, may 2015.

MOSHEIOV, G. V-shaped policies for scheduling deteriorating jobs. **Operations Research**, 1991. Institute for Operations Research and the Management Sciences (INFORMS), v. 39, n. 6, p. 979–991, dec 1991.

MOSHEIOV, G.; ORON, D. Due-window assignment with unit processing-time jobs. **Naval Research Logistics**, 2004. Wiley, v. 51, n. 7, p. 1005–1017, 2004.

MOSHEIOV, G.; ORON, D. Due-date assignment and maintenance activity scheduling problem. **Mathematical and Computer Modelling**, 2006. Elsevier BV, v. 44, n. 11-12, p. 1053–1057, dec 2006.

MOSHEIOV, G.; SARIG, A. A due-window assignment problem with position-dependent processing times. **Journal of the Operational Research Society**, 2008. Informa UK Limited, v. 59, n. 7, p. 997–1003, jul 2008.

MOSHEIOV, G.; SARIG, A. Scheduling a maintenance activity and due-window assignment on a single machine. **Computers & Operations Research**, 2009. Elsevier BV, v. 36, n. 9, p. 2541–2545, sep 2009.

MOSHEIOV, G.; SARIG, A. Scheduling identical jobs and due-window on uniform machines. **European Journal of Operational Research**, 2010. Elsevier BV, v. 201, n. 3, p. 712–719, mar 2010.

MOSHEIOV, G.; SARIG, A. A note: a due-window assignment problem on parallel identical machines. **Journal of the Operational Research Society**, 2011. Informa UK Limited, v. 62, n. 1, p. 238–241, jan 2011.

MOSHEIOV, G.; SIDNEY, J. B. Scheduling with general job-dependent learning curves. **European Journal of Operational Research**, 2003. Elsevier BV, v. 147, n. 3, p. 665–670, jun 2003.

MOSHEIOV, G.; YOVEL, U. Minimizing weighted earliness–tardiness and due-date cost with unit processing-time jobs. **European Journal of Operational Research**, 2006. Elsevier BV, v. 172, n. 2, p. 528–544, jul 2006.

NEARCHOU, A. C. A differential evolution approach for the common due date early/tardy job scheduling problem. **Computers & Operations Research**, 2008. Elsevier BV, v. 35, n. 4, p. 1329–1343, apr 2008.

NG, C. T. D. et al. Single machine scheduling with a variable common due date and resource-dependent processing times. **Computers & Operations Research**, 2003. Elsevier BV, v. 30, n. 8, p. 1173–1185, jul 2003.

PANWALKAR, S.; RAJAGOPALAN, R. Single-machine sequencing with controllable processing times. **European Journal of Operational Research**, 1992. Elsevier BV, v. 59, n. 2, p. 298–302, jun 1992.

PANWALKAR, S. S.; SMITH, M. L.; SEIDMANN, A. Common due date assignment to minimize total penalty for the one machine scheduling problem. **Operations Research**, 1982. Institute for Operations Research and the Management Sciences (INFORMS), v. 30, n. 2, p. 391–399, apr 1982.

PARSA, N. R.; KARIMI, B.; HUSSEINI, S. M. Exact and heuristic algorithms for the just-in-time scheduling problem in a batch processing system. **Computers & Operations Research**, 2017. Elsevier BV, v. 80, p. 173–183, apr 2017.

PEREIRA, J.; VASQUEZ, O. C. The single machine weighted mean squared deviation problem. **European Journal of Operational Research**, 2017. Elsevier BV, v. 261, n. 2, p. 515–529, sep 2017.

PLATEAU, M.-C.; RIOS-SOLIS, Y. Optimal solutions for unrelated parallel machines scheduling problems using convex quadratic reformulations. **European Journal of Operational Research**, 2010. Elsevier BV, v. 201, n. 3, p. 729–736, mar 2010.

QUADDUS, M. A. A generalized model of optimal due-date assignment by linear programming. **Journal of the Operational Research Society**, 1987. Informa UK Limited, v. 38, n. 4, p. 353–359, apr 1987.

RABADI, G.; ANAGNOSTOPOULOS, G. C.; MOLLAGHASEMI, M. A heuristic algorithm for the just-in-time single machine scheduling problem with setups: a comparison with simulated annealing. **The International Journal of Advanced Manufacturing Technology**, 2007. Springer Nature, v. 32, n. 3-4, p. 326–335, mar 2007.

RABADI, G.; MOLLAGHASEMI, M.; ANAGNOSTOPOULOS, G. C. A branch-and-bound algorithm for the early/tardy machine scheduling problem with a common due-date and sequence-dependent setup time. **Computers & Operations Research**, 2004. Elsevier BV, v. 31, n. 10, p. 1727–1751, sep 2004.

RAGHAVACHARI, M. A v-shape property of optimal schedule of jobs about a common due date. **European Journal of Operational Research**, 1986. Elsevier BV, v. 23, n. 3, p. 401–402, mar 1986.

RIBAS, I.; COMPANYS, R.; TORT-MARTORELL, X. An iterated greedy algorithm for the flowshop scheduling problem with blocking. **Omega**, 2011. Elsevier BV, v. 39, n. 3, p. 293–301, jun 2011.

RIBAS, I.; COMPANYS, R.; TORT-MARTORELL, X. An iterated greedy algorithm for solving the total tardiness parallel blocking flow shop scheduling problem. **Expert Systems with Applications**, 2019. Elsevier BV, v. 121, p. 347–361, may 2019.

RIOS-SOLIS, Y. A.; SOURD, F. Exponential neighborhood search for a parallel machine scheduling problem. **Computers & Operations Research**, 2008. Elsevier BV, v. 35, n. 5, p. 1697–1712, may 2008.

ROCHOLL, J.; MÖNCH, L. Hybrid algorithms for the earliness–tardiness single-machine multiple orders per job scheduling problem with a common due date. **RAIRO - Operations Research**, 2018. EDP Sciences, v. 52, n. 4-5, p. 1329–1350, oct 2018.

ROCHOLL, J.; MÖNCH, L. Decomposition heuristics for parallel-machine multiple orders per job scheduling problems with a common due date. **Journal of the Operational Research Society**, 2019. Informa UK Limited, p. 1–17, aug 2019.

RODRIGUEZ, F. J. et al. An iterated greedy algorithm for the large-scale unrelated parallel machines scheduling problem. **Computers & Operations Research**, 2013. Elsevier BV, v. 40, n. 7, p. 1829–1841, jul 2013.

RONCONI, D. P.; KAWAMURA, M. S. The single machine earliness and tardiness scheduling problem: lower bounds and a branch-and-bound algorithm. **Computational & Applied Mathematics**, 2010. Brazilian Society for Computational and Applied Mathematics (SBMAC), v. 29, n. 2, jun 2010.

RUIZ, R.; STÜTZLE, T. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. **European Journal of Operational Research**, 2007. Elsevier BV, v. 177, n. 3, p. 2033–2049, mar 2007.

SAKURABA, C. S.; RONCONI, D. P.; SOURD, F. Scheduling in a two-machine flowshop for the minimization of the mean absolute deviation from a common due date. **Computers & Operations Research**, 2009. Elsevier BV, v. 36, n. 1, p. 60–72, jan 2009.

SARPER, H. Minimizing the sum of absolute deviations about a common due date for the two-machine flow shop problem. **Applied Mathematical Modelling**, 1995. Elsevier BV, v. 19, n. 3, p. 153–161, mar 1995.

SCULLI, D.; TSANG, K. Priority dispatching rules in a fabrication/assembly shop. **Mathematical and Computer Modelling**, 1990. Elsevier BV, v. 13, n. 3, p. 73–79, 1990.

SHI, H.-B.; WANG, J.-B. Research on common due window assignment flowshop scheduling with learning effect and resource allocation. **Engineering Optimization**, 2019. Informa UK Limited, p. 1–18, may 2019.

SIDNEY, J. B. Optimal single-machine scheduling with earliness and tardiness penalties. **Operations Research**, 1977. Institute for Operations Research and the Management Sciences (INFORMS), v. 25, n. 1, p. 62–69, feb 1977.

SRIDHARAN, V.; ZHOU, Z. A decision theory based scheduling procedure for single-machine weighted earliness and tardiness problems. **European Journal of Operational Research**, 1996. Elsevier BV, v. 94, n. 2, p. 292–301, oct 1996.

SRIRANGACHARYULU, B.; SRINIVASAN, G. An exact algorithm to minimize mean squared deviation of job completion times about a common due date. **European Journal of Operational Research**, 2013. Elsevier BV, v. 231, n. 3, p. 547–556, dec 2013.

STÜTZLE, T.; RUIZ, R. Iterated greedy. In: **Handbook of Heuristics**. [S.l.]: Springer International Publishing, 2018. p. 547–577.

SU, L.-H. Minimizing earliness and tardiness subject to total completion time in an identical parallel machine system. **Computers & Operations Research**, 2009. Elsevier BV, v. 36, n. 2, p. 461–471, feb 2009.

SUN, H.; WANG, G. Parallel machine earliness and tardiness scheduling with proportional weights. **Computers & Operations Research**, 2003. Elsevier BV, v. 30, n. 5, p. 801–808, apr 2003.

SUNDARARAGHAVAN, P. S.; AHMED, M. U. Minimizing the sum of absolute lateness in single-machine and multimachine scheduling. **Naval Research Logistics**, 1984. Wiley, v. 31, n. 2, p. 325–333, jun 1984.

SUNG, C.; MIN, J. Scheduling in a two-machine flowshop with batch processing machine(s) for earliness/tardiness measure under a common due date. **European Journal of Operational Research**, 2001. Elsevier BV, v. 131, n. 1, p. 95–106, may 2001.

SURIYAARACHCHI, R. H.; WIRTH, A. Earliness/tardiness scheduling with a common due date and family setups. **Computers & Industrial Engineering**, 2004. Elsevier BV, v. 47, n. 2-3, p. 275–288, nov 2004.

SZWARC, W. Single machine scheduling to minimize absolute deviation of completion times from a common due date. **Naval Research Logistics**, 1989. Wiley, v. 36, n. 5, p. 663–673, oct 1989.

SZWARC, W. The weighted common due date single machine scheduling problem revisited. **Computers & Operations Research**, 1996. Elsevier BV, v. 23, n. 3, p. 255–262, mar 1996.

T'KINDT, V.; SHANG, L.; CROCE, F. D. Exponential time algorithms for just-in-time scheduling problems with common due date and symmetric weights. **Journal of Combinatorial Optimization**, 2019. Springer Science and Business Media LLC, v. 39, n. 3, p. 764–775, dec 2019.

TOKSARI, M. D.; GUNER, E. Minimizing the earliness/tardiness costs on parallel machine with learning effects and deteriorating jobs: a mixed nonlinear integer programming approach. **The International Journal of Advanced Manufacturing Technology**, 2008. Springer Science and Business Media LLC, v. 38, n. 7-8, p. 801–808, jul 2008.

TOKSARI, M. D.; GUNER, E. Parallel machine earliness/tardiness scheduling problem under the effects of position based learning and linear/nonlinear deterioration. **Computers & Operations Research**, 2009. Elsevier BV, v. 36, n. 8, p. 2394–2417, aug 2009.

TOKSARI, M. D.; GUNER, E. The common due-date early/tardy scheduling problem on a parallelmachine under the effects of time-dependent learning and linear and nonlinear deterioration. **Expert Systems with Applications**, 2010. Elsevier BV, v. 37, n. 1, p. 92–112, jan 2010.

TOKSARI, M. D.; GUNER, E. Parallel machine scheduling problem to minimize the earliness/tardiness costs with learning effect and deteriorating jobs. **Journal of Intelligent Manufacturing**, 2010. Springer Science and Business Media LLC, v. 21, n. 6, p. 843–851, mar 2010.

VENTURA, J. A.; WENG, M. X. An improved dynamic programming algorithm for the single-machine mean absolute deviation problem with a restrictive common due date. **Operations Research Letters**, 1995. Elsevier BV, v. 17, n. 3, p. 149–152, apr 1995.

VENTURA, J. A.; WENG, M. X. Single machine scheduling with a common delivery window. **Journal of the Operational Research Society**, 1996. Informa UK Limited, v. 47, n. 3, p. 424–434, mar 1996.

WANG, J.-B. Single machine scheduling with common due date and controllable processing times. **Applied Mathematics and Computation**, 2006. Elsevier BV, v. 174, n. 2, p. 1245–1254, mar 2006.

WANG, J.-B.; WANG, C. Single-machine due-window assignment problem with learning effect and deteriorating jobs. **Applied Mathematical Modelling**, 2011. Elsevier BV, v. 35, n. 8, p. 4017–4022, aug 2011.

WANG, J.-B.; WANG, M.-Z. Single machine multiple common due dates scheduling with learning effects. **Computers & Mathematics with Applications**, 2010. Elsevier BV, v. 60, n. 11, p. 2998–3002, dec 2010.

WANG, J.-B.; WANG, M.-Z. Single-machine due-window assignment and scheduling with learning effect and resource-dependent processing times. **Asia-Pacific Journal of Operational Research**, 2014. World Scientific Pub Co Pte Lt, v. 31, n. 05, p. 1450036, oct 2014.

WANG, J.-B. et al. Due-window assignment scheduling problems with position-dependent weights on a single machine. **Engineering Optimization**, 2020. Informa UK Limited, v. 52, n. 2, p. 185–193, mar 2020.

WEBSTER, S.; JOG, D.; GUPTA, A. A genetic algorithm for scheduling job families on a single machine with arbitrary earliness/tardiness penalties and an unrestricted

common due date. **International Journal of Production Research**, 1998. Informa UK Limited, v. 36, n. 9, p. 2543–2551, sep 1998.

WEEKS, J. K.; FRYER, J. S. A methodology for assigning minimum cost due-dates. **Management Science**, 1977. Institute for Operations Research and the Management Sciences (INFORMS), v. 23, n. 8, p. 872–881, apr 1977.

WENG, M. X.; VENTURA, J. A. Scheduling about a large common due date with tolerance to minimize mean absolute deviation of completion times. **Naval Research Logistics**, 1994. Wiley, v. 41, n. 6, p. 843–851, oct 1994.

WENG, M. X.; VENTURA, J. A. A note on "common due window scheduling". **Production and Operations Management**, 1996. Wiley, v. 5, n. 2, p. 194–200, jan 1996.

WENG, M. X.; VENTURA, J. A. Single-machine earliness-tardiness scheduling about a common due date with tolerances. **International Journal of Production Economics**, 1996. Elsevier BV, v. 42, n. 3, p. 217–227, apr 1996.

WENG, W.; FUJIMURA, S. Self evolution algorithm to minimize earliness and tardiness penalties with a common due date on a single machine. **IEEJ Transactions on Electrical and Electronic Engineering**, 2008. Wiley, v. 3, n. 6, p. 604–611, nov 2008.

WILAMOWSKY, Y.; EPSTEIN, S.; DICKMAN, B. Optimal common due-date with completion time tolerance. **Computers & Operations Research**, 1996. Elsevier BV, v. 23, n. 12, p. 1203–1210, dec 1996.

WU, Y.; LAI, K. K. A production scheduling strategy with a common due window. **Computers & Industrial Engineering**, 2007. Elsevier BV, v. 53, n. 2, p. 215–221, sep 2007.

XIAO, W.-Q.; LI, C.-L. Approximation algorithms for common due date assignment and job scheduling on parallel machines. **IIE Transactions**, 2002. Informa UK Limited, v. 34, n. 5, p. 466–477, may 2002.

XIONG, X. et al. Single-machine scheduling and common due date assignment with potential machine disruption. **International Journal of Production Research**, 2017. Informa UK Limited, v. 56, n. 3, p. 1345–1360, jul 2017.

YANG, S.-J. Single-machine scheduling problems with both start-time dependent learning and position dependent aging effects under deteriorating maintenance consideration. **Applied Mathematics and Computation**, 2010. Elsevier BV, v. 217, n. 7, p. 3321–3329, dec 2010.

YANG, S.-J.; YANG, D.-L.; CHENG, T. C. E. Single-machine due-window assignment and scheduling with job-dependent aging effects and deteriorating maintenance. **Computers & Operations Research**, 2010. Elsevier BV, v. 37, n. 8, p. 1510–1514, aug 2010.

YEUNG, W.; OGUZ, C.; CHENG, T. Single-machine scheduling with a common due window. **Computers & Operations Research**, 2001. Elsevier BV, v. 28, n. 2, p. 157–175, feb 2001.

YEUNG, W.; OĞUZ, C.; CHENG, T. E. Two-stage flowshop earliness and tardiness machine scheduling involving a common due window. **International Journal of Production Economics**, 2004. Elsevier BV, v. 90, n. 3, p. 421–434, aug 2004.

YI, Y.; WANG, D. W. Soft computing for scheduling with batch setup times and earliness-tardiness penalties on parallel machines. **Journal of Intelligent Manufacturing**, 2003. Springer Science and Business Media LLC, v. 14, n. 3/4, p. 311–322, 2003.

YIN, Y. et al. Single-machine batch delivery scheduling with an assignable common due window. **Omega**, 2013. Elsevier BV, v. 41, n. 2, p. 216–225, apr 2013.

YIN, Y. et al. Common due date assignment and scheduling with a rate-modifying activity to minimize the due date, earliness, tardiness, holding, and batch delivery cost. **Computers & Industrial Engineering**, 2012. Elsevier BV, v. 63, n. 1, p. 223–234, aug 2012.

YIN, Y. et al. Four single-machine scheduling problems involving due date determination decisions. **Information Sciences**, 2013. Elsevier BV, v. 251, p. 164–181, dec 2013.

YING, K.-C. Minimizing earliness–tardiness penalties for common due date single-machine scheduling problems by a recovering beam search algorithm. **Computers & Industrial Engineering**, 2008. Elsevier BV, v. 55, n. 2, p. 494–502, sep 2008.

YING, K. C.; LIN, S. W.; HUANG, C. Y. Sequencing single-machine tardiness problems with sequence dependent setup times using an iterated greedy heuristic. **Expert Systems with Applications**, 2009. Elsevier BV, v. 36, n. 3, p. 7087–7092, apr 2009.

YING, K.-C.; LIN, S.-W.; LU, C.-C. Effective dynamic dispatching rule and constructive heuristic for solving single-machine scheduling problems with a common due window. **International Journal of Production Research**, 2017. Informa UK Limited, v. 55, n. 6, p. 1707–1719, aug 2017.

YOUSEFI, M.; YUSUFF, R. M. Minimising earliness and tardiness penalties in single machine scheduling against common due date using imperialist competitive algorithm. **International Journal of Production Research**, 2013. Informa UK Limited, v. 51, n. 16, p. 4797–4804, aug 2013.

YUAN, J. A note on the complexity of single-machine scheduling with a common due date, earliness-tardiness, and batch delivery costs. **European Journal of Operational Research**, 1996. Elsevier BV, v. 94, n. 1, p. 203–205, oct 1996.

YUCE, B. et al. Hybrid genetic bees algorithm applied to single machine scheduling with earliness and tardiness penalties. **Computers & Industrial Engineering**, 2017. Elsevier BV, v. 113, p. 842–858, nov 2017.

ZHANG, X. et al. Single-machine common/slack due window assignment problems with linear decreasing processing times. **Engineering Optimization**, 2016. Informa UK Limited, v. 49, n. 8, p. 1388–1400, nov 2016.

ZHAO, C.; TANG, H. A note to due-window assignment and single machine scheduling with deteriorating jobs and a rate-modifying activity. **Computers & Operations Research**, 2012. Elsevier BV, v. 39, n. 6, p. 1300–1303, jun 2012.

ZHAO, H.; LI, G. Unbounded batch scheduling with a common due window on a single machine. **Journal of Systems Science and Complexity**, 2008. Springer Nature, v. 21, n. 2, p. 296–303, feb 2008.

ZHU, H. et al. Due-window assignment and scheduling with general position-dependent processing times involving a deteriorating and compressible maintenance activity. **International Journal of Production Research**, 2015. Informa UK Limited, v. 54, n. 12, p. 3475–3490, jul 2015.

ZHU, Z. et al. Due-window assignment and scheduling with multiple rate-modifying activities under the effects of deterioration and learning. **Mathematical Problems in Engineering**, 2011. Hindawi Limited, v. 2011, p. 1–19, 2011.