Robust and stable optimization for parallel machine scheduling problems

Widad Naji

Communauté
UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTE UNIVERSITE GRENOBLE ALPES

Spécialité : **Génie Industriel**

Arrêté ministériel : 25 mai 2016

Présentée par

« Widad NAJI »

Thèse dirigée par **Van-Dat CUNG, Professeur, Grenoble INP**, et codirigée par **Marie-Laure Espinouse, Professeur, Université de Grenoble Alpes** préparée au sein du **Laboratoire G-SCOP** dans **l'École Doctorale IMEP2 (Ingénierie - Matériaux Mécanique Energétique Environnement Procédés Production)**

Robust Optimization and Stability Analysis for Parallel Machine Scheduling

Optimisation robuste et analyse de stabilité pour les problèmes d'ordonnancement sur machines parallèles

Thèse soutenue publiquement le **«02 Mai 2018»**,
devant le jury composé de :

Monsieur André ROSSI
Professeur à l'Université d'Angers, Rapporteur
Monsieur Éric SANLAVILLE
Professeur à l'Université du Havre, Rapporteur
Monsieur Alexandre DOLGUI
Professeur à l'IMT Atlantique, Président
Monsieur Roel LEUS
Professeur à KU Leuven, Examinateur
Madame Marie-Laure ESPINOUSE
Professeur à l'Université de Grenoble Alpes, Co-directrice de thèse
Monsieur Van-Dat CUNG
Professeur à Grenoble INP, Directeur de thèse

Acknowledgements

First and foremost, I wish to thank my advisors: Mrs M.-L. Espinouse, Professor at the University of Grenoble and Mr V.-D. Cung, Professor at Grenoble Institute of Technology. They have been supportive since I began working on this topic in my master thesis. Ever since, they have supported me not only by providing a research assistantship over almost four years, but also emotionally through the rough road to finish this thesis. They helped me come up with the thesis topic and guided me over almost four year of development. And during the most difficult times when writing this thesis, they gave me the moral support I needed to move on. I would like to express my deep gratitude to them. During my years of PhD at G-SCOP Laboratory, I had the chance to work in a pleasant atmosphere. Therefore, I would like to thank my colleagues and all the staff in G-SCOP Laboratory.

I wish also to express all my thanks to the committee members for the honour they gave me by participating to my thesis committee. I am indebted to Mr A. Rossi, Professor at the University of Angers, and Mr E. Sanlaville, Professor at the University le Havre, for accepting to report my thesis manuscript, for all the constructive remarks and valuable corrections. I am also indebted to my examiners Mr A. Dolgui, Professor at the IMT Atlantique and Mr R. Leus, Professor at KU Leuven for letting my defense be an enjoyable moment and for their brilliant comments and suggestions.

Finally, I would like to acknowledge my parents, my brother and my closest friends. Thank you very much for your unconditional love and support. Thank you for being my guiding light, for encouraging me, and for making me who I am today. I am extremely grateful for everything that you do.

Contents

List of Figures

List of Tables

General introduction

In the nineteenth century, the thinking was characterized by the certainty. It was marked by a confidence in the power of science and technology. Newtonian physics, for instance, showed that the universe functions according to rigid laws and that its course could be predicted with total confidence and certainty. But, a scientific revolution began to take shape when the Quantum Theory and the Theory of Relativity proved that the knowledge was incomplete since there as places in the universe such as black holes from which no information at all could be obtained. Chaos Theory also showed the limits of predicting or controlling the world. Thus, the twentieth century was marked by the emergence of uncertainty and the fall of certainty. This new perception of the world has made a revolutionary effect not only in physics, but in all disciplines: economy, decision-making, policy, computational modelling and simulation, *etc.*.

In computational modelling, optimization and simulation, uncertainty is inevitable when we consider real-world problems, systems or phenomena. In this discipline, the uncertainty faced by the modellers can basically arise from the model form or from the empirical data (See Morgan and Henrion (1990) classification in Appendix A). The model form uncertainty can be defined as an inadequacy in the representation of the problem due to simplifications and reasoning approximations. Sometimes, it can also arise from the difficulty to choose among alternative models. But, as a model is a simplified representation intended to enhance our ability to understand, simulate, predict and control a system, a perfect compliance between the model and the reality never exists in an absolute sense. In the other hand, the model form uncertainty can be overcome only through model validation. Once the model form is validated, the model's usefulness will depend on high part on the accuracy of the data. In fact, data uncertainty can make the exploitation of the optimization and simulation results far from the desired goal in most of real-world applications.

In scheduling, the gap between the results provided by the classical scheduling algorithms and their applicability in production environment requirements was pointed out by both researchers and practitioners. One of the reasons of such a gap is connected with the assumptions that data (related to jobs and/or machines) are known with certainty. However, scheduling in production environment is recognized to be a complex and dynamic activity in which job processing times can take longer or shorter than forecast, release dates and due dates may change, jobs may have to be inserted or canceled. Besides, the resources may become unavailable due to unforeseen events as breakdowns, accidents, operator absenteeism, *etc.*. Assuming a certain data can make the classical scheduling approach inadequate to solve scheduling problems under uncertainty. Several publications claimed that ignoring data uncertainty can lead the decision-maker to be confronted with infeasible or sub-optimal schedules. Thus, computing an optimal schedule has become less important than protecting the schedule from uncertainty. We should recognize that we are no longer at the beginning of the road to develop approaches and models for dealing with optimization problems under uncertainty in

general and scheduling ones in particular. The number of published works in this issue exploded, but the number of problems affected with data uncertainty is unlimited. Besides, there is neither a unique approach nor a unique model to deal with uncertainty arising in a given problem. For these reasons, optimization under uncertainty provides fertile ground for researchers.

Thesis objectives and contributions: In this thesis, we focus on the makespan minimization on parallel machines with splitting (*resp.* preemption) under uncertain processing times. Parallel Machine Scheduling (PMS) consists in sequencing n jobs on m same function machines in order to optimize the objective. The processing time of a job on a machine represents the duration required by this machine to process this job.

PMS is a very common problem in many manufacturing industries and also in multiprocessor computing applications. In PMS, we distinguish three environment machines: the identical, the uniform, and the unrelated parallel machines. In the identical parallel machines, the job processing time is independent of the machine where it is processed. In the uniform parallel machines, each machine has a different speed and the processing times of a job in two different machines are proportional. In unrelated parallel machines, no particular relationship exists between the processing times on the different machines, *i.e.*, there is no proportionality between the processing time of a job on a given machine and its processing time on another machine. From a mathematical point of view, the unrelated parallel machines represent the general case of PMS.

The splitting and the preemption often take place in problems. Indeed, we consider that we can divide each job into sub-jobs that can be processed independently on the machines. When we accept to overlap the sub-jobs of the same job, we address the PMS under job splitting otherwise we address the preemptive PMS. The makespan minimization on parallel machines with job splitting is a relaxed version of the preemptive problem. In environments that are not affected by uncertainty, the makespan minimization on parallel machines under job splitting (*resp.* preemption) can be solved in polynomial time. Indeed, the most general case which is the makespan minimization on preemptive unrelated parallel machines has been shown to be polynomial by Lawler and Labetoulle (1978). However, the uncertainty of processing times is prevalent in such problems. When we consider this uncertainty, the deterministic models can lead to schedules with poor performance when applied to the actual realizations of processing times. Therefore, we investigate the makespan minimization on parallel machines, under both splitting and preemption, taking into account the uncertain processing times. Thus, our contributions can be described in response to the following research questions:

Research question 1: How to provide solutions that withstand the uncertain processing times?

To reach this goal, we adopt a proactive approach under which we anticipate the uncertainty in order to provide robust solutions. We consider then, that in a lot of activities, the experts are able to describe uncertain data realizations by the construction of a set of discrete scenarios representing several potential futures. But, in our problems, a subsequent difficulty is related to the feasibility of the constraints under all the scenarios. We initially propose two different approaches to deal with the uncertain processing times. The first approach aims to enforce the feasibility of the schedule under different scenarios by tolerating the violations of some processing requirements. Slack variables are used to compensate such a loss. In the second approach, we

propose an artificial scenario solution based algorithm. Under this approach, we construct a set of feasible solutions and we choose among them the most robust. This choice is based on an evaluation algorithm that computes a robustness measure under each solution. Thanks to this second approach, we respect the hard constraints and ensure their feasibility under all the scenarios without any violations. However, the artificial scenario solutions are not optimal. Therefore, we propose a third approach in which we use the min-max objective -widely used in robust optimization- to generate optimal robust schedules.

The second question that we propose to answer through this thesis is related to the robustness cost of the solutions. Indeed, several authors and practitioners highlight that robustness has a cost: the more you robustify the solution, the more you deteriorate the performance and consequently the more you pay, and vice versa. In this thesis, we propose to quantify the cost of robustness of each solution in order to verify this postulate. We formulate this question as follows:

Research question 2: What is the robustness cost when we use each of the computed robust solutions?

To answer this question, we define a robustness cost indicator and then we analyse and compare the evolution of the robustness cost of the robust solutions according to different parameters: the problem size, the discrete scenario set size and the degree of uncertainty. Indeed, we distinguish low uncertainty degree, medium uncertainty degree and high uncertainty degree.

When we answer the previous questions, new research questions arise:

Research question 3: When the future turns out differently than considered, does a robust solution guarantee a good performance under the new scenario? And how much adjustment is needed to be also protected against this new scenario?

To answer these questions, we propose to evaluate the stability of the robust solutions under new scenarios. We evaluate both the performance stability and the structure stability.

Thesis outline: This thesis is divided into six chapters, providing an introduction, the major contributions and an outlook on further research questions and perspectives.

Chapters 1 and 2 provide the materials to the understanding of the issue of optimization under uncertainty in general and scheduling under uncertainty in particular. In chapter 1, we provide a background for the understanding of the various aspects of data uncertainty. We explore the existing concepts and classifications of uncertainty from different fields. Then, we propose a unified taxonomy of uncertainty in computational modeling and simulation. We propose a synthesized view of uncertainty type representations. We also survey their related optimization approaches and models. At the end, we propose a general methodology to deal with data uncertainty in optimization problems. In chapter 2, we focus on scheduling. We present the general form of the scheduling problem and define its fundamental notions. We discuss the limitations of scheduling under the hypothesis of certainty and we show the necessity to take into account uncertainty. Besides, we briefly review the representations of uncertainty in scheduling. And then, we survey the existing approaches to deal with uncertainties and disruptions when the objective is robustness and/or stability. In light of this, we show that globally there exist a lot of opportunities for future developments and we identify some areas for future research. Regarding our research questions, we identify the key tracks for the development of clear answers to these questions.

Chapters 3, 4, 5 and 6 contain the major ideas and contributions of this thesis. In Chapter 3, we focus on the makespan minimization on unrelated parallel machines under splitting and preemption respectively. We present these problems in the static deterministic cases and stress the need to consider processing time uncertainty. We provide the main motivations for our work by examples. Therefore, we show that the optimal solutions computed based on a unique scenario lead to schedules with poor performance when the processing times turn to be different from the forecast one. Then, we propose to model the uncertainties by considering a set of potential scenarios (discrete scenario representation) and we show that the uncertainty in these problem models is affecting the equality constraints which impacts the feasibility. As initial contributions, we propose two different approaches to deal with the uncertainty of processing times. The first approach aims to enforce the feasibility of the schedule under different scenarios by tolerating the violations of some processing requirements. In the second approach, we propose an artificial scenario based approach. Under this approach, we construct a set of feasible solutions and we choose the robust solutions based on an evaluation algorithm that compute the robustness measures of each solution. We do extensive computational experiments to test the validity of the third approach in both splitting and preemption cases. These results are serving to a comparable basis in the next chapters.

In Chapter 4, we show that the direct application of robust optimization techniques leads to no solution. To overcome this difficulty, we propose formulations in which the uncertainty appears in the objective instead of the assignment constraints. Thus, we provide robust solutions in polynomial time, and we extend these results to the special cases of uniform and identical parallel machines. Extensive numerical experiments are carried out to illustrate these new proposals. We extend the application of the robust discrete optimization to the preemptive unrelated parallel machines under discrete processing time scenarios in Chapter 5. A major difference between the preemptive case and the splitting one is due to the computation of a sequence. We consider that the decision-maker is interested in computing a unique assignment solution under the set of discrete scenarios and that he accepts to have different sequences. Here again, we provide robust assignment solutions in polynomial time while the sequence construction remains the same as in the deterministic case. The same test protocol is applied to evaluate the robustness cost of the solutions.

Finally, Chapter 6 is devoted to evaluating the stability of the structure and the performance of the robust solutions when adding a new scenario. In this approach, the term stability is used for the algorithm step at which robust solutions of the problem have been already computed, and additional calculations are performed in order to investigate how these solution structures and performances depend on changes in the set of potential scenarios. In the experimental results, we focus on the splitting case.

Chapter 1

Uncertainty: Conceptual classifications, Representations and Solving Approaches

Abstract: The goal of this chapter is to provide a background for the understanding of the various aspects of uncertainty. We introduce the existing concepts and classifications of uncertainty from different fields. In light of this, we propose a unified taxonomy of uncertainty in computational modeling and simulation. Besides, we review the uncertainty models developed by the mathematical community and we classify them according to their potential to represent the uncertainty types. Then, we focus on the approaches and models to formulate and solve optimization problems under uncertainty. We identify the main objectives of optimization models under uncertainty and propose a cross classification of the optimization models under uncertainty according to the objectives and to the uncertainty types. Lastly, we provide a guide to optimization under uncertainty.

Introduction

Data uncertainties are various. When they affect a modeling and simulation problem, they may require different modeling techniques and elicit divergent resolution approaches. The uncertainty conceptual classifications the most considered in the literature are based on two dimensions which are the type and the level. By the type of uncertainty, we mean its whatness *i.e.* the nature resulting from its inherent characteristics. And by the level of uncertainty, we mean its rank between full knowledge and total ignorance, if it is closer to certainty or to ignorance, and where it can be situated between them. In Section1, we review some of these conceptual taxonomies provided by different communities. The objective of this section is to highlight the convergences between existing conceptual taxonomies. Indeed, we show that there exists a consensus about uncertainty types contrary to what it is often thought. Moreover, we show that the uncertainty type and the uncertainty level classifications are highly linked. Based on this review, we propose a unified taxonomy of uncertainty considering a type-level dimension. In Section2, we review the models (formal and practical) developed by the mathematical community in order to represent the uncertainties. We show that the uncertainty models can also be classified according to their potential to represent adequately each uncertainty type. In Section3, we address the modeling approaches and formulations to solve a decision problem under uncertainty. Given that each optimization problem under uncertainty may require a particular approach, we introduce a guide to choose the adequate model based on different entries.

1.1 Uncertainty conceptual classifications

1.1.1 Uncertainty type classification

The economist philosophy introduced by Knight (1921) and Keynes (1936) was the first to build the modern history of uncertainty. Motivated by the need to develop a realistic economic theory under which the full knowledge does not exist.

***a) Uncertainty vs Risk*:**

Knight (1921) introduced the notion of uncertainty in a distinct sense of the notion of risk that was familiar to the economists at that time. Knight (1921) considered that uncertainty refers to situations where the probability of future events is unknown and immeasurable while risk refers to situations where future events occur with measurable probability. For Knight (1921), the probability distribution under uncertainty is unknown because the uncertain events are unique, they can not be reducible to a group of similar cases which does not allow the use of probabilities contrarily to risk under which probabilities could either be theoretically identified or determined from empirical frequencies. Knight (1921) called this distinction as the *'practical difference'* between risk and uncertainty and he wrote that *"the practical difference between the two categories, risk and uncertainty, is that in the former the distribution of the outcome in a group of instances is known (either through calculation a priori or from statistics of past experience), while in the case of uncertainty this is not true, the reason being in general that it is impossible to form a group of instances, because the situation dealt with is in a high degree unique."*

Keynes (1936) defended similar ideas about uncertainty based upon the impossibility of determining accurate probabilities. Keynes (1936) affirmed that uncertainty concerns basically future events that are influenced by various changes (economical, social, political) which make their extrapolation from the actual events impossible: *"By uncertain knowledge, let me explain, I do not mean merely to distinguish what is known for certain from what is only probable. The game of roulette is not subject, in this sense, to uncertainty... The sense in which I am using the term is that in which the prospect of a European war is uncertain, or the price of copper and the rate of interest twenty years hence... About these matters, there is no scientific basis on which to form any calculable probability whatever. We simply do not know."*

Knight (1921) and Keynes (1936) classifications appeared to be very similar. However, the interpretations of their texts emphasized that when talking about missing probabilities, Keynes and Knight have something very different in mind. In Knight (1921), the decision-maker lack of information to construct a probability distribution while in Keynes (1936) such a probability does not exist in a non-ergodic world. Many debates and analyses were constructed around these interpretations ((See for example the analysis of Davidson (1996)).

b) Risk and Savage's uncertainty vs Ambiguity:

By the mid-20^{th} century and under the influence of *'the theory of games and economic behavior'* developed by Von Neumann and Morgenstern (1944) and *'the equilibrium'* of Nash *et al.* (1950), a new uncertainty classification has emerged which is about the environmental uncertainty and the behavioral uncertainty. Environmental uncertainty is when the management of a firm has little information about its environment that is unpredictable while behavioral uncertainty arises from the difficulty to predict the actions of other relevant actors in a multi-actor system, particularly in view of the potential for opportunistic ((See Koopmans (1957) and Williamson (1993) for more discussion). Indeed, if decisions are made in a social context, the data of the problem may be determined by the decisions made by other actors who are in situations similar to our situation.

Following this, the future is unknown and decisions must be taken under uncertainty because probability can base only on individual's subjective estimates or what is called subjective probabilities. Therefore, Savage (1954) claimed that objective probabilities which are related to a randomness of the real world do not exist, decision-makers can only assign subjective probabilities reflecting their degrees of belief to the events. Savage's uncertainty has been used to describe subjective risk under which the probabilities are "personal" as termed in Savage (1954). He proved that, if the decision-maker adheres to the axioms of rationality, believing an uncertain event has possible outcomes each with a utility function, then the decision -maker choices can be explained as arising from the combination of this utility function and the subjective probability attached to each outcome. Under this subjectivist viewpoint of uncertainty, he made the implication that all uncertainties can be reduced to risk. But, Ellsberg (1961) and Shackle *et al.* (1972), who were among the first economists to insist on the importance of uncertainty, rejected the use of numerical probabilities (even subjective) for representing the uncertainty in situations where it is impossible to distinguish meaningfully between the relative "likelihood" of the outcomes. Relying on the psychology literature, Ellsberg (1961) chose the term ambiguity to describe the situations under which it might not be desirable to behave according to the postulate of Savage.

c) Ambiguity Vs Fundamental uncertainty:

A new refinement of the uncertainty concept has been given by Dequech (2000) who noticed that the Keynesian uncertainty refers not only to ambiguity, but also to fundamental uncertainty without any distinction between them. Therefore, he defined the fundamental uncertainty as a significant indeterminacy in the future that reflects situations under which not only decision makers cannot attach probabilities to the future events but a scenario that is not completely imagined may occur. Accordingly, the fundamental uncertainty is characterized by the possibility of creativity, innovation and structural change in a dynamic context which is not the case under ambiguity ((See Freeman (1982), Davidson (1991), Dosi and Egidi (1991)). Furthermore, Dequech (2000) argued that fundamental uncertainty is not synonymous with ignorance since its degrees vary according to many factors as human creativity and technological innovations, in contrary to ignorance that is absolute. Dow (2016) proposed a diagrammatic representation of the different understanding of uncertainty according to the economist philosophies.

The economist classifications have been very influential, especially among decision-making and management community. Based on the Keynesian classification, Luce and Raiffa (1957) divided decision situations into three classes based on whether it has been carried out under conditions of certainty, risk or uncertainty. They distinguished: decision under certainty, decision under risk, and decision under uncertainty. This classification is discussed and used until now to precise the context under which a decision has to be made (Armbruster and Delage (2015), Kochenderfer *et al.* (2015)). The risk was refereed in the early papers of global optimization by Dantzig *et al.* (1955) while the vagueness or fuzziness dates back to Zadeh (1965) who means by decision-making in a fuzzy environment, a decision process in which the goals and/or the constraints constitute classes of alternatives whose boundaries are not sharply defined. In which concerns the ambiguity, this later was addressed by Kouvelis and Yu (1997).

d) Aleatory uncertainty Vs Epistemic uncertainty:

Since the 90's, the risk analysis community has manifested a growing interest to the definition of uncertainty and to the classification of uncertainty types. The risk analysis community has distinguished mainly two natures of uncertainty which are the aleatory uncertainty and the epistemic uncertainty ((See for instance: Hammonds *et al.* (1994), Helton (1994), Rowe (1994), Hora (1996), Ferson and Ginzburg (1996), Bedford and Cooke (2001), Der Kiureghian and Ditlevsen (2009)). The aleatory uncertainty is defined as the inherent variation associated with the physical system or the environment under consideration while the epistemic uncertainty is defined as the lack of knowledge or information. The epistemic uncertainty could be reduced by enhancing the state of knowledge and not the aleatory uncertainty. The refinement of this classification by different communities, through the works of Smithson (1989, 2012), Klir and Wierman (1998), Oberkampf *et al.* (2004), Ayyub and Klir (2006) and Wierman (2010), revealed a consensus about three distinct types of uncertainty in spite of the use of different semantics. These uncertainty types are the aleatory uncertainty, the ambiguity and the vagueness ((See definitions and synonyms in table 1.1 and figure 1.1).

In behavioral sciences, Smithson (1989, 1990, 2012) divided uncertainty into three types which are the probability, the non specificity (ambiguity) and the vagueness ((See their typology and definitions in appendix B table B.1). A much closed taxonomy to those of Smithson (1989) has been developed in engineering and sciences by Ayyub and Klir (2006). Their taxonomy also reported in Appendix B table B.3,

FIGURE 1.1: Uncertainty types identified by different communities

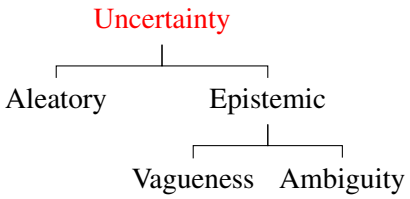


TABLE 1.1: Uncertainty type meanings and synonyms

Type	Definition	Synonyms
Aleatory	Inherent variation associated with the physical system or the environment (Oberkampf *et al.* (2004))	- likelihood (Ayyub and Klir (2006)), - probability (Smithson (1989)), - randomness or frequency (Ayyub and Klir (2006)),
Vagueness	Information imprecisely defined, or indistinct (Oberkampf *et al.* (2004))	- imprecision or approximation (Ayyub and Klir (2006)), - fuzziness (Wierman (2010))
Ambiguity	Multiplicity of alternatives without preference to any alternative Wierman (2010)	- non-specificity Smithson (1990)

considered that uncertainty can be a likelihood, an ambiguity or an approximation. Klir and Wierman (1998) and Wierman (2010) whose research was about the development of measures of uncertainty in mathematical systems also identified ambiguity and vagueness to be two distinct types of uncertainty ((See the appendix B figure and table B.4). Accordingly, ambiguity deals with the multiplicity without preference to an alternative, we (See multiple concepts but what is (Seen does not allow to identify precisely the concept (Seek. Vagueness deals with lack of definite or sharp distinction, we (See one concept, but it does not have the resolution to determine its identity or its frontier. Oberkampf *et al.* (2004) considered that the epistemic uncertainty includes the vagueness, the ambiguity ((See the appendix B figure and table B.5).

1.1.2 Uncertainty level classification

Walker *et al.* (2003) defined uncertainty to be *"any departure from the unachievable ideal of complete determinism"* where determinism reflects the situations where we know exactly everything. He used the level of uncertainty as a dimension for its classification. Accordingly, he divided the spectrum between certainty and total ignorance to three intermediate levels of uncertainty. These three intermediate levels are: L1- Statistical uncertainty, L2- Scenario uncertainty, L3 Recognized ignorance.

Statistical uncertainty concerns uncertainty that can be described adequately in statistical terms. Scenario uncertainty is when a range of possible alternatives is known, but it is not possible to describe the probability distribution of any of them. And recognized ignorance is when ignoring the statistical properties and we are unable to develop scenario representation. Walker's level dimension has been used as a starting point for new uncertainty classifications in different applications. Courtney (2003), Makridakis and Taleb (2009), Kwakkel *et al.* (2010), Walker *et al.* (2010), and Walker *et al.* (2013)) identified five levels of uncertainty including

TABLE 1.2: Uncertainty levels according to the economist risk analysis community

Uncertainty level	Meaning
L1	Decision-maker admits that he is not absolutely certain, but he is able to measure the degree of uncertainty explicitly
L2	Statistical uncertainty (Walker *et al.* (2003))
L3	Decision-maker is able to enumerate multiple alternatives and is able to rank them in term of perceived likelihood.
L4	Scenario uncertainty (Walker *et al.* (2003)): Decision-maker is able to enumerate multiple alternatives and is not able to rank them in term of perceived likelihood.
L5	Recognized ignorance (Walker *et al.* (2003))

the levels identified by Walker ((See Table 1.2). Walker *et al.* (2013) referred to scenario uncertainty and recognized ignorance as *"deep uncertainty"*.

The levels of uncertainty appeared also in the economy. Dequech (2011) has proposed a typology of the main concepts of uncertainty used by economists in which he has introduced indirectly the uncertainty levels. Indeed, he has distinguished between weak and strong uncertainty. Under weak uncertainty, the decision-maker can form a unique, additive, and fully reliable probability distribution. In contrast, strong uncertainty is characterized by the absence of such a distribution. According to this, Dequech (2011) combines the levels with the uncertainty types. He considered that both weak and strong uncertainties are composed of two subcategories each. The first subcategory is the Keynesian risk while the second is Savage's uncertainty. The first subcategory of strong uncertainty is ambiguity while the second is fundamental uncertainty.

Likewise, in computational modeling and simulation, the levels of uncertainty are found basically in manufacturing systems, a classification which is about known uncertainties, suspicions about the future and complete unknowns has a large spread ((See McKay and Wiers (1999), Vieira *et al.* (2003), Aytug *et al.* (2005*a*), Wojakowski and Warżołek (2014)). In this classification the known uncertainties are those for which some information is available, the suspicions about the future arise from the intuition and the experience of the human scheduler and unknowns are unpredictable events.

In the next section, we propose to initiate a unified taxonomy of uncertainty based on the reviewed literature. For this purpose, we have to answer these questions:

- Is there a consensus about uncertainty types? And is there a link between uncertainty types and uncertainty levels?

1.2 Toward a unified taxonomy of data uncertainty

1.2.1 The consensus about uncertainty types

In the timeline represented in Figure E.17, we represent the evolution of the uncertainty concept through the different communities cited above. We consider three categories: the blue color represents the economists, the purple color represents the decision-making community while the green color represents the scientific communities (risk analysis, engineering, *etc.*), and orange for behavioral science. From the timeline, we

remark a first consensus about two distinct types of uncertainty which are the aleatory uncertainty and the ambiguity. The aleatory uncertainty is termed risk by the economists and decision-making community, probabilistic, statistical or likelihood by the others. This reveals that aleatory uncertainty is of two distinct types which are: the objective aleatory uncertainty under which an objective probability distribution exists, and the subjective aleatory uncertainty under which only a subjective probability distribution can be constructed based on expert's belief. They are both characterized by the existence of an underlying probability distribution. The ambiguity definitions are often related to the multiplicity of alternatives without a preference to anyone, the lack of knowledge under ambiguity is related to the probability of each alternative.

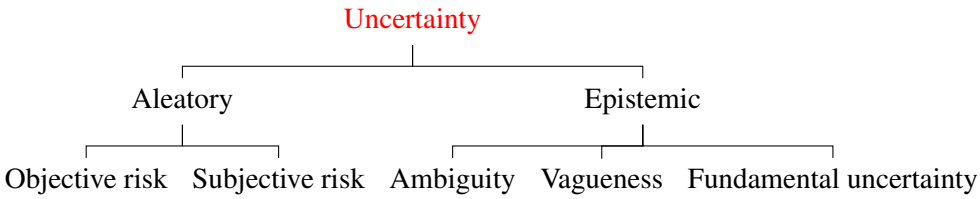
Indeed, We can also remark a second consensus from the different communities except the economists, about the vagueness (fuzziness) as a distinct type. Ambiguity is then characterized by the lack of knowledge. Consequently, ambiguity is an epistemic uncertainty.

The second consensus that completes the first one considers besides ambiguity another epistemic uncertainty type which is vagueness (*i.e.* fuzziness). This type of uncertainty was defined by all the communities except the economists. The definition of fuzziness is basically related to the imprecision. The lack of knowledge under vagueness is related to the frontier that allows to make a sharp distinction.

The economist community identified another type which is the fundamental uncertainty defined as the impossibility to build any scenario, the lack of knowledge under fundamental uncertainty is related to the information that allow to construct a scenario because of a dynamic nature of the environment.

As a result, we can consider that uncertainty is of 5 types (objective risk, subjective risk, ambiguity, vagueness and fundamental uncertainty) that can be classified according to two categories (aleatory uncertainty and epistemic uncertainty) as given in Figure 1.2.

FIGURE 1.2: A unified taxonomy of uncertainty types



1.2.2 The link between uncertainty types and uncertainty levels

The level classification given by Dequech (2011) showed that the type and the level of uncertainty in economy are linked ((See Table 1.3).

The same observation can be made for the level classification, in Table 1.4, proposed by Walker *et al.* (2003), Courtney (2003), Makridakis and Taleb (2009), Kwakkel *et al.* (2010), Walker *et al.* (2010), and Walker *et al.* (2013). Indeed, when we analyze the classification that was given by Walker *et al.* (2003), we deduce that the statistical uncertainty is linked to the objective risk, the scenario uncertainty is linked to ambiguity and the recognized ignorance is associated to the fundamental uncertainty. In the improved

TABLE 1.3: Uncertainty types and levels according to the economist philosophy

level	Associated type
Weak	Aleatory
	Subjective risk
Strong	Ambiguity
	Fundamental uncertainty

TABLE 1.4: Uncertainty types and levels according to risk analysis community

Level		Associated type
L1	Decision-makers is not absolutely certain, but he is able to measure the degree of uncertainty explicitly	
L2	Statistical uncertainty (Walker *et al.* (2003))	Aleatory (objective risk)
L3	Decision-maker enumerate multiple alternatives rank them in term of perceived likelihood	Aleatory (subjective risk)
L4	Scenario uncertainty (Walker *et al.* (2003))	Ambiguity
L5	Recognized ignorance (Walker *et al.* (2003))	Fundamental uncertainty

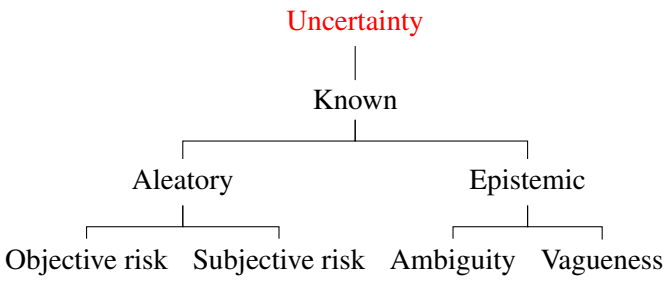
classification given by Walker *et al.* (2013), we can also link the fourth level of uncertainty with the subjective risk.

Based on this, the level dimension and the type dimension are linked. These uncertainty types represent different level of severity, consequently we consider that the data uncertainty type dimension is correlated to what is called the level dimension.

1.2.3 A unified taxonomy of data uncertainty

In the modeling, optimization and simulation problems under uncertainty, the different uncertainties need to be represented with different models and require divergent solving approaches. From a modeling point of view, the type dimension is useful as it permits to choose an adequate representation of the uncertain data, and from a decision-making point of view considering the level (severity) is necessary to choose an appropriate approach to deal with the uncertainty. Therefore, we propose a taxonomy in which the uncertainties are classified into types that are ranked according to levels.

FIGURE 1.3: Known uncertainty types



We consider at first that the spectrum of data knowledge can be divided into three domains: a) certain data, b) known uncertainties, c) unknowns.

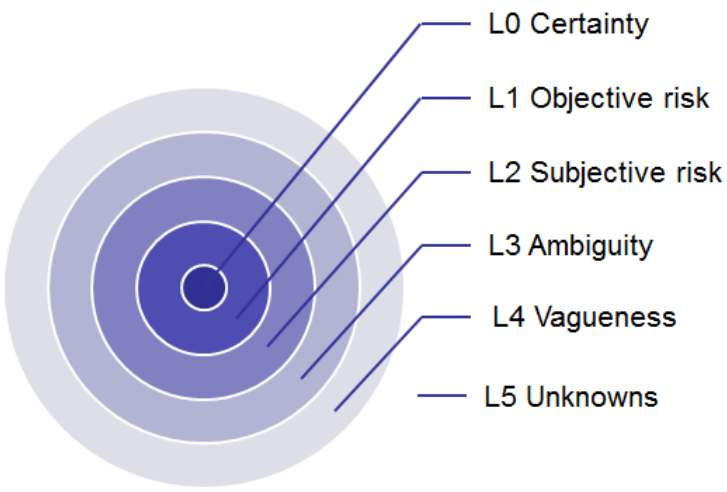
Data is certain when the knowledge about it is complete. Known uncertainties are uncertain data that experts/modelers are able to describe somehow by using historical information or by relying on their belief degree. And the unknowns are uncertainties that cannot be described, in this state, values that are not at all imagined could be taken by the uncertain data because unimaginable events may occur (for example in situations of crisis or natural disaster).

Based on these definitions, the aleatory uncertainty, the ambiguity and the vagueness belong to known uncertainties(Figure 1.3) while fundamental uncertainty belongs to unknowns. Consequently, fundamental uncertainty represents the highest level of uncertainty that we denote L5 while the known uncertainties can be classified according to four levels:

- L1 uncertainty corresponds to objective risk: probability distribution is available.
- L2 uncertainty corresponds to subjective risk: multiple alternatives with attached probabilities.
- L3 uncertainty corresponds to ambiguity: multiple alternatives without attached probabilities.
- L4 uncertainty to vagueness: the boundaries between alternatives are imprecise.

The progressive transition from certainty (that we classify at L0) to unknowns (that we classify at L5) can be described through a diagrammatic representation in which the types of uncertainty are ranked as given in Figure 1.4.

FIGURE 1.4: The progressive transition from certainty to unknowns



In the next section, we focus on the modeling of data uncertainties.

1.3 Uncertainty representations

The literature offers a variety of models to represent uncertainty types. These models can be classified into two distinct categories: formal models, and practical models. Formal representations gather all the mathematical theories that require effort and time for development, they are used in particular in theoretical disciplines and in applications that relate to economy, science and engineering (*e.g.* Dubois and Prade (2009), Wierman (2010)). Practical representations gather all the models that provide satisfactory representation of the uncertain data (Halpern (2005)), they are used in particular in risk analysis, decision-making and management sciences, and computational and simulation community (*e.g.* planning and scheduling, transportation and logistics, *etc.*).

Each of these models is developed to represent a particular type of uncertainty. In this section, we present a brief survey of the formal and practical models of uncertainty with regard to their potential to represent each uncertainty type.

1.3.1 Formal uncertainty representations

The study of probability dates back to Pascal and Fermat in the 17^{th} century, but the complete axiomatic foundation of probability theory was given by Kolmogorov (1933). The theory of probability is adequate to represent probabilistic uncertainty. But, the probability has two interpretations. The first interpretation of probability is the frequentist interpretation called objective probability, where the probability is the limit on the frequency of the event when the experiment is repeated many times (Bernouilli's law of large numbers). The second interpretation of the probability is when the notion of repeatability of an experiment is excluded. The probability is then a subjective value quantifying the belief that the expert gives to the occurrence of an event (Ramsey (1931), De Finetti (1937), Savage (1954)).

Dempster-Shafer theory of evidence (Dempster (1967) and Shafer *et al.* (1976)), possibility theory has been identified as attractive alternatives to model belief when the probability distribution is not known. These theories are primarily adequate to model epistemic uncertainty. The common point between these theories is to present the uncertainty by a family of probability distributions and not by a single probability distribution since this unique probability distribution is hard to identify. Thus, the idea is to build from the available data, two functions as an envelope containing the unknown distribution. The difference between the two functions reflects the uncertainty.

The theory of evidence is considered as a generalization of the theory of subjective probabilities in that it is based on less restrictive assumptions on the measures of belief, with the idea to use the only available information to reason. It is fundamentally based on the use of the mass function (or mass distribution) and two non-additive measures, belief and plausibility functions. The belief function takes into account all the available evidence. During the same period, Zadeh (1965) has introduced the theory of possibilities based on fuzzy set theory. Fuzzy set theory provides using necessity and possibility measures to express belief degree. An expert may believe that a given parameter is within a certain range and may even have an intuitive feel for the best value within that range.

For an extensive review on formal representations of uncertainty, the reader is referred to Dubois and Prade (2009).

1.3.2 Practical uncertainty representations

Almost all practical representations are based on scenario model. Rockafellar and Wets (1991) considered scenario model as a common approach to model uncertainty in practice:" The uncertainty about parameters of the system is modeled by a small number of versions of sub-problems derived from an underlying optimization problem. These correspond to different scenarios a word that is used to suggest some kind of limited representation of information on the uncertain elements or how such information may evolve." The major motivation to choose the models based on scenarios is to get relatively a modest number of constructed representative instances so that the equivalent problem can be computationally tractable.

Kouvelis and Yu (1997) claimed that scenario model is one of the important tools in structuring data uncertainty when multiple future alternatives are potentially realizable without any attached probabilities. The scenario model requires the construction of a set of instances or data sets containing numerical values that will represent several contrasting futures (economic, technological and business possibilities of tomorrow).

Beside the purpose of modeling future alternatives, scenario model is also suitable to model objective risk (*e.g.* Nowak and Tomasgard (2007), Schütz *et al.* (2009)). Indeed, for the sake of simplicity, a discretization of probability distribution can lead to the construction of discrete scenarios with attached probabilities. For example, if there are three random variables with six outcomes, and the random variables are independent, we can build 6x6x6=864 scenarios with attached probabilities. To reduce the scenario tree to a manageable size, Monte Carlo simulation is commonly used. In some other situations, especially when dealing with subjective risk, it is possible to construct a tree of scenarios by eliciting expert's opinion about the future alternatives and their attached probabilities. The interval representation is also widely used in practice. It is better known as bounded uncertainty representation (Ben-Tal and Nemirovski (2000); Lin *et al.* (2004)). The interval represents the range of all possible realizations of the uncertain data. This model is useful when only the upper and lower bounds can be determined with certainty and the decision-maker is not able to choose specific scenarios. It is a continuous set representation.

The polyhedral sets based representations also represent a practical tool to model uncertainty. The uncertain data vary within a specific set. For instance, d_0 is the nominal value of the uncertain data and Z is the user specified primitive uncertainty set:

$$d = d_0 + \sum_{l=1}^{L} \zeta_l d_l : \zeta \in Z \tag{1.1}$$

The geometry of the convex set Z leads to different uncertainty sets as box set, ellipsoidal set, polyhedral set, *etc.* ((See Ben-Tal *et al.* (2009)).

The main objective of these representations is to bring back the uncertain problem into a deterministic one. The focus on polyhedral set representations is mainly related to the tractability of the mathematical models contaminated with the uncertainty of data. They do not increase the complexity of the problem *i.e.*

for some of the most important convex optimization problems (linear optimization particularly). For more discussions on the choice of the uncertainty sets, the reader is referred to Li and Floudas (2014) and Gorissen *et al.* (2015).

1.3.3 Uncertainty representations and types relationship matrix

From the reviewed literature, we notice that each type of uncertainty (aleatory, ambiguity and vagueness) can be represented adequately with at least one formal and one practical representations (Table 1.5).

TABLE 1.5: Uncertainty representations and uncertainty types relationship matrix

	Objective risk	**Subjective risk**	**Ambiguity**	**Vagueness**
Objective Probability	x			
Subjective Probability		x		
Possibility theory				x
Evidence theory			x	
Scenarios			x	
Weighted scenarios	x	x		
Intervals			x	
Fuzzy intervals				x
Polyhedral sets	x	x	x	

Objective probability can be used to describe objective risk. For instance, it is frequent to consider that the demand uncertainty is random and to represent its probability distribution (*e.g.* Liao and Rittscher (2007), You *et al.* (2011),Wang *et al.* (2015), Ji *et al.* (2016)).

The subjective probability, called also *Bayesian probability* was introduced to model subjective risk (Savage's uncertainty). For example, when managers have only some partial information about the demand distribution in addition to demand observations, they cannot consider the demand as a random variable (*e.g.* Saghafian and Tomlin (2016)) but, they could assign subjective probabilities to the demand based on their beliefs. Nevertheless, the fact that additive axiom is still considered even under subjective probabilities makes this theory not really adequate for modeling belief degree (Colyvan (2008), Liu (2015)). In accordance with the axiom of additivity, if we believe that demand will increase with a probability equal to 60%, we should automatically attribute a probability equal to 40% to the fact that demand will decrease even if we do not have any belief about demand decrease. When we use belief functions based on the evidence, the 60% reflects the level of belief that demand will increase, but no belief that demand will decrease is expressed thus the belief that 'demand will decrease' is equal to 0% while 40% is undecided, it may pertain to 'the demand will increase' or to 'the demand will decrease' depending on the additional evidence. Based on the belief, the plausibility that demand will increase is 100% (60% level of belief comes from the direct evidence and 40% level of belief comes from the ambiguity), and the plausibility that demand will decrease is 40%. Thus, the belief function measure is more conservative than the one obtained using the probabilistic framework. Accordingly, the theory of evidence is suitable to model ambiguity.

Scenario model is suitable to model ambiguity. Under ambiguity, the uncertain demand can be represented by a vector of potential demands: $d \in \{d_1, d_2, ..., d_n\}$. Under objective or subjective risk, the aleatory

demand can be represented by a vector of potential demand scenarios with attached probabilities p_i: $d \in \{(d_1, p_1), (d_2, p_1), ..., (d_n, p_n)\}$. These probabilities represent the weights of scenarios. Interval model can be considered as a scenario representation under which the number of scenarios is infinite. Therefore, we can consider that interval model can also be suitable to model ambiguity but with a continuous set. For example, if an expert can express its uncertainty as "demand will not be more than $\overline{d}$ and not less than $\underline{d}$ but I do not know exactly which value it will take", the most appropriate model is the interval model: $d \in [\underline{d}, \overline{d}]$.

The theory of possibility provides the appropriate framework to model vagueness (*i.e.* fuzziness). For instance, an expert may believe that a given parameter is within a certain range and may even have an intuitive feel for the best value within that range. For example, if demand can be vaguely expressed in different linguistic terms like 'demand is about d_m but *definitely* not less than d_l and not greater than d_u'. Then, the demand is a fuzzy variable with triangular possibility distribution where d_m is the most possible value that definitely belongs to the set of available values. Another example is when the decision maker expresses the vagueness as 'demand is *much larger* than d_l. The reader is referred to Petrovic *et al.* (1999) for more insight on the use of fuzzy sets to model demand uncertainty.

The polyhedral sets based representations are appropriate models to represent ambiguity (Ben-Tal *et al.* (2009), Bertsimas and Sim (2003)). Also, in the case of aleatory uncertainty, some authors claimed that there are probabilistic arguments allowing to uncertainty polyhedral sets to represent the random uncertainty (*e.g.* Bertsimas and Brown (2009); Bandi and Bertsimas (2012)).

1.4 Optimization under uncertainty: Approaches and models

Optimization is a central issue to a number of disciplines as production and logistics, engineering design, policy-making and many others. We speak about optimization under uncertainty when we have a problem involving data uncertainty with one (or more) objective(s) to optimize. The basic idea of optimization under uncertainty is to integrate the available information about the uncertain data into the optimization problem formulation in order to compute solutions that withstand the uncertainty with a performance guarantee. The literature distinguishes mainly two optimization approaches under uncertainty: stochastic optimization where uncertainty data is assumed to follow a known probability distribution and robust optimization approach where uncertain data has an unknown probability distribution ((See Kouvelis and Yu (1997)). Besides, Sahinidis (2004) considered fuzzy optimization as a proactive optimization approach.

1.4.1 Stochastic optimization approach and models

Stochastic optimization approach assumes that the uncertainty has a probabilistic description. In the stochastic optimization, three modeling variants are studied: the stochastic recourse models without control, the controlled stochastic recourse models and the chance constrained models. Recourse models often apply to situations in which decisions are made repeatedly in essentially the same circumstances, and the goal is to compute a solution that will perform well on average. The philosophy behind recourse stochastic models is to take some actions in the first stage of optimization. Then, when a random event occurs and affects

the outcome of the first-stage decision, recourse decisions can be made in the following stages as corrective actions to compensate any bad effects that might have been experienced as a result of the first-stage decision. Stochastic models with recourse are widely used in multi-stage decision problems. The most applied and studied multi-stage stochastic recourse models are the two-stage linear programs. The treatment of these models and their resolution methods can be found in (Birge and Louveaux (2011); Van Slyke and Wets (1969); Ruszczynski and Shapiro (2003)). Sahinidis (2004) and Mulvey *et al.* (1995) considered that the two stage stochastic optimization model does not account for the variability of the second stage costs and might lead to solutions where the actual second-stage costs are unacceptably high. Indeed, a decision is based on a first-stage and expected second-stage costs with the assumption that the decision-maker is risk-neutral. To capture the notion of risk in stochastic optimization, Mulvey *et al.* (1995) integrate goal optimization formulations with a scenario-based description of problem data. Their robust stochastic optimization model formalizes a way to measure the trade-off between feasibility and optimality and enforce a restricted recourse. Mulvey *et al.* (1995) proposed the minimization of the variance of the second stage decision variables. But, the variance is a symmetric risk measure penalizing costs, both above and below the expected recourse cost equally. Besides, it made the problem non linear. Therefore, Vladimirou and Zenios (1997) restricted the dispersion of the second-stage solutions to a prescribed level ϵ while Ahmed and Sahinidis (1998) introduced an index that is asymmetric, which measures only costs that are higher than the expected one and insures the linearity of the problem. Kuhn *et al.* (2011) and Georghiou *et al.* (2015) also proposed special restricted recourse to multi-stage stochastic optimization problems. In chance constrained models, the focus is put on the reliability of the constraints which is the ability to meet feasibility in an uncertain environment. This reliability is expressed as a minimum requirement on the probability of satisfying constraints. There is often uncertainty regarding the constraint matrix, and the system is required to satisfy the corresponding constraint with a fixed probability ((See Charnes and Cooper (1959), Kall (1976)).

1.4.2 Robust optimization approach and models

In the robust optimization approach, the uncertainty model is not stochastic, but rather deterministic and set-based. The objective of robust models is to provide solutions that are feasible for all the realizations of the uncertain data, and that perform well in the worse-case. They use min-max and min-max regret objectives that were originally defined in the strategic decision-making literature by Gupta and Rosenhead (1968),Rosenhead *et al.* (1972*a*), and introduced later in operations research problems by Rosenblatt and Lee (1987). But, the issue of robustness can be historically traced back to robust control concepts developed in the early 30s by Bode and others, in the context of feedback amplifiers. The robust control policy is static rather than adapting to measurements of variations, the controller is designed to work assuming that variables are uncertain but bounded. Questions such as the stability margin, which is the amount of feedback gain required to destabilize a controlled system, led naturally to a "worst-case" point of view, in which bad parameter values are too dangerous to be allowed, even with low probability.

The robust optimization models date back to Soyster (1973). But, the large part of robust optimization literature has been developed since the publication of the works of Mulvey *et al.* (1995); Kouvelis and Yu (1997); Ben-Tal and Nemirovski (1998); El Ghaoui *et al.* (1998) in the late 1990's. Those works have

developed many min-max and min-max regret models along with tractable solving techniques for both practical and polyhedral set uncertainty representations, especially in linear cases. This has made robust models computationally more attractive than stochastic models ((See Ben-Tal *et al.* (2009),Bertsimas and Sim (2004),Averbakh (2006)). The basic robust optimization approach paradigm was based on three assumptions formulated in Ben-Tal *et al.* (2009). First, all variables should have values as a result of solving the problem before the actual data reveals itself. Secondly, the decision maker is fully responsible for the consequences of the decisions to be made when, and only when, the actual data are within the pre-specified uncertainty set. And finally, the constraints of the uncertain problem in question are hard *i.e.*, the decision maker cannot tolerate violations of constraints when the data are in the pre-specified uncertainty set. Robust models deal with hard constraints that cannot be relaxed while stochastic models deal with soft constraints since we can tolerate the infeasibility of some constraints in the first optimization stage ((See Mulvey *et al.* (1995); Kouvelis and Yu (1997); Ben-Tal *et al.* (2009); Bertsimas *et al.* (2011)). But, in the recent development of robust optimization models, the multi-stages notion has been introduced to robust optimization. Therefore, the first assumption has been relaxed: the first-stage decisions are made before the uncertainty is revealed and the second stage decisions can be adjusted later according to the actual data. Robust optimization was extended to a multi-stage setting and to random data sets, where the recourse decisions are adapted to the realization of the random data as they unfold in stages. Nemirovski *et al.* (2009) proposed to restrict the recourse decisions to be affine dependent on the random parameters. This approach could perform reasonably well and even achieve optimality ((See for example, Bertsimas *et al.* (2010) and Iancu and Trichakis (2013)). The robust discrete optimization, introduced by Mulvey *et al.* (1995); Kouvelis and Yu (1997), is a framework of robust optimization under which the uncertainty representation is scenario based. It has been applied as a way of self-protection in many industrial applications against undesirable impacts due to uncertain data. Decisions made according to min-max, min-max regret are the most used whenever the uncertain data are modeled using discrete scenarios.

1.4.3 Fuzzy optimization approach and models

Fuzzy optimization approach dates back to the seminal work by Bellman and Zadeh (1970) who founded the basis of decision-making in a fuzzy environment. The spread of fuzzy optimization models dates back to Zimmermann (1978, 2010). Fuzzy optimization models are useful when the problems under consideration include vaguely defined relationships, or human evaluations Kacprzyk and Orlovski (2013). In these models, uncertain data are considered as fuzzy numbers and constraints are treated as fuzzy sets. The degree of satisfaction of a constraint is defined as the membership function of the constraint and some constraint violation is allowed. Objective functions in fuzzy optimization are treated as constraints. The membership function is used to represent the degree of satisfaction of constraints, the decision-maker expectations about the objective function level, and the range of uncertainty of data. Flexible optimization models and possibilistic optimization models are two variants of fuzzy optimization. The former considers uncertainty only in the constraint coefficients while the second recognizes uncertainties in the objective function coefficients as well as in constraint coefficients. For a comprehensive review on fuzzy optimization, the reader is referred to Sahinidis (2004) and Lodwick and Untiedt (2010).

In the next section we propose a general methodology for optimization under data uncertainty. In this methodology, we introduce a global reasoning under data uncertainty.

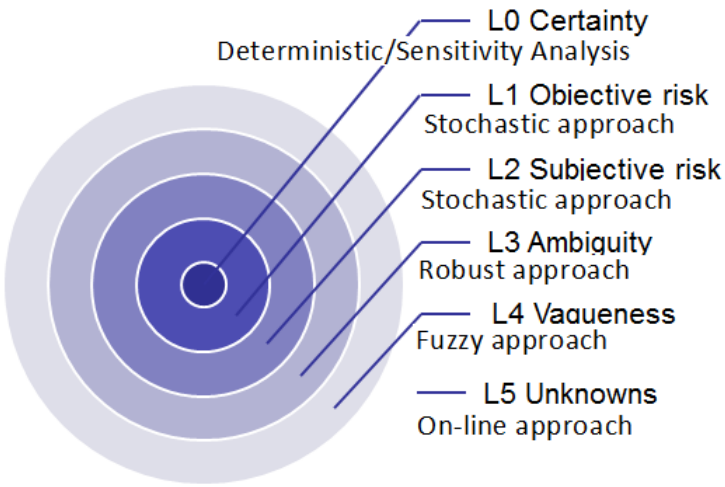
1.5 A general framework for optimization under data uncertainty

The general framework for optimization under data uncertainty can be described through five major steps ((See Figure 1.6):

Step 1 (COLLECT AND ANALYZE) aims to collect and analyze the available knowledge about the uncertain data based on existing databases and expert opinions.

Step 2 (TYPIFY) aims to the identification of the uncertainty type and level dealt with based on the analysis of the available knowledge about the uncertain data. And, ***Step 3 (REPRESENT UNCERTAINTY)*** aims to represent the uncertain data based on the analysis and typification. In steps 2 and 3, we can be aided by the unified taxonomy given in Figure 1.3 and the adequacy matrix between uncertainty types and uncertainty representations provided in Table 1.5.

FIGURE 1.5: Optimization approaches under data uncertainty



STEP 4 (SELECT APPROACH) aims, in parallel to step 3, to select an appropriate optimization approach based on the data uncertainty analysis and the environment requirements. This choice can be guided by the uncertainty level classification. According to the level of uncertainty, we can distinguish:

- Deterministic optimization approach when one considers that the data is certain (L0)
- Sensitivity analysis when we are certain but to go further we analyse the sensitivity of the deterministic solution under small variations of data (complementary to the deterministic approach under L0).
- Proactive optimization approaches where we anticipate known uncertainties (Stochastic for L1 and L2, Robust for L3 and Fuzzy for L4).

- And online optimization approaches under unknowns, here the decision is constructed in a dynamic way as the unknown data become known (under L5).

Step 5 (VERIFY) aims to verify of the adequacy between the optimization approach and the uncertainty type-level ((See Figure 1.5). And finally, ***Step 6 (MODEL AND SOLVE)***, in this step the choice of the optimization model is related to two components: the objective and the uncertainty type (objective risk, subjective risk, ambiguity, fuzziness). The review of the optimization approaches and models in Section1.4 reveals 5 distinct objectives. Under these objectives the decision- maker aims to provide:

- A solution that insures a reliability over a given threshold or,
- A solution that performs well on average, without control of the corrective actions or,
- A solution that performs well on average, with control of the corrective actions or,
- A solution that performs well in the worst-case or,
- A solution to adapt in case conditions change or,
- A solution that satisfies fuzzy constraints.

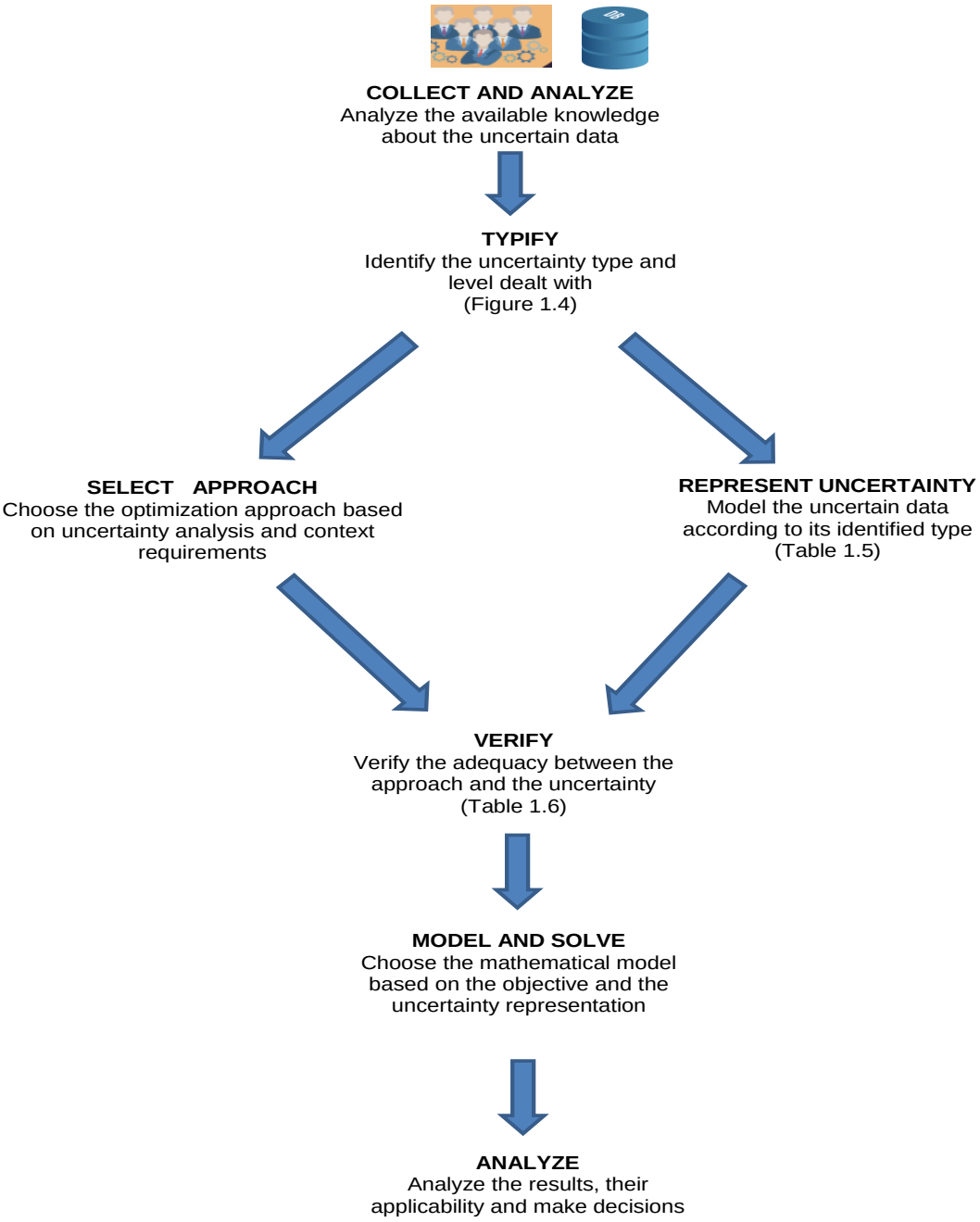
Therefore, the existing optimization models can be classified as follows:

- Chance constrained models to deal with risk whenever the objective is to provide a reliable solution.
- Stochastic recourse models to deal with the aleatory uncertainty whenever the objective is to come up with a solution that perform well in average without any control of the corrective actions.
- Stochastic models with restricted recourse to deal with risk whenever the objective is to come up with a solution that perform well on average with control of the corrective actions.
- Robust stochastic models to deal with risk whenever the objective is to come up with a solution that perform well in the worst-case.
- Robust static models to deal with ambiguity whenever the objective is to come up with a solution that perform well in the worst-case without any changes.
- Robust adjustable models to deal with ambiguity whenever the objective is to come up with a solution that perform well in the worst-case and prepare to adapt in case conditions change.
- Fuzzy models to deal with vagueness and to come up with a solution that satisfies fuzzy (imprecise) constraints.

For each model or formulation, there exists prescribed rules for resolution that are developed in the literature.

The last step is ***STEP 7 (ANALYZE)***, it is common to all optimization problems and it aims to analyze the results, to test their applicability.

FIGURE 1.6: General framework for optimization under data uncertainty



Conclusion

In this chapter, we reviewed some conceptual classifications of uncertainty based on two dimensions: the type and the level of uncertainty. We showed through the discussed taxonomies that a consensus about uncertainty types exists between different communities. Besides, we showed that the type and the level dimensions are linked. Therefore, we proposed a unified taxonomy of uncertainty according to a type-level dimension. In this taxonomy we distinguished four uncertainty types which are: the objective risk, the subjective risk, the ambiguity and the vagueness. Each uncertainty type reflects an uncertainty level. Besides, we reviewed the models developed by the mathematical community in order to represent the different uncertainty types and we classify these models according to their potential to represent the uncertainty types. Then, we addressed the mathematical approaches and models to formulate and solve a decision problem under uncertainty. We identified the main objectives in optimization under uncertainty and proposed to classify the optimization models under uncertainty, according to these objectives and to the uncertainty types. The organization of these informations built the general methodology for optimization under uncertainty.

In the next chapter, we will focus on scheduling under uncertainty. We aim to review how uncertainty is considered in scheduling problems, what are the approaches and the objectives of scheduling under uncertainty.

Chapter 2

Robustness and Stability in scheduling under uncertainty

Abstract: The goal of this chapter is to provide an overview of the state of the art in scheduling under uncertainty. We present the general form of the scheduling problem and define its fundamental notions. We discuss the limitations of the classical scheduling approach under uncertainty and disruptions. Furthermore, we review the representations of the uncertainties or disruptions faced in scheduling and we survey the existing approaches to deal with them. Then, we focus on robustness and stability issues in scheduling under uncertainty. We survey the existing contributions in robust scheduling, and scheduling for stability purpose. In light of this, we show that there exist a lot of opportunities for future works in this area.

Contents

Introduction

Scheduling is an optimization problem that aims to allocate resources to jobs over time in order to optimize one or more objectives under given constraints (Pinedo (2012)). A large variety of algorithms -exact or approximate- are designed to solve scheduling problems, but the vast majority of these algorithms is based on the assumption of a predictable environment where information about data are complete and are not subject to change. They generate a predictive schedule, assuming a deterministic, certain and static environment. However, manufacturing environments are subject to numerous sources of uncertainty as demand and cost fluctuations, process variability, resources reliability/availability, *etc.*. Under these uncertainties, an optimal predictive schedule with respect to the deterministic model and the initial data will turn to be suboptimal or infeasible during the execution. Therefore, an increasing attention was paid to scheduling problems under uncertainty in both academic and industrial research areas. The objective is to provide robust solutions that schedule jobs in such a way as to use the resources efficiently under uncertain processing times.

In this chapter, we review the existing approaches to deal with uncertainty in scheduling, and we focus on the proactive scheduling approach. We survey approaches and models to provide the robustness and the stability.

2.1 Scheduling Generalities

2.1.1 General form of the scheduling problem

Scheduling is an optimization problem that aims to allocate and sequence jobs on resources over time in order to optimize one or more objectives under given constraints. In its most general form, the scheduling problem is defined as follows:

Given
-a set J of n jobs that must be executed,
-a set M of m resources to process the activities,
-a set of constraints which must be satisfied, and
-a set of objectives to judge a schedule's performance,

what is the best way to assign and sequence the jobs on the resources over time such that all of the constraints are satisfied and the best performance is reached?

A *job* $j \in J$ consists of a number of tasks. If a job j is composed of only one task, then we denote p_j the processing requirement of j. Furthermore, a job may have a release date r_j on which the job became available for processing, a due date d_j which is the date in which the job is promised to be finished, a starting time s_j which is the effective date in which a job starts its processing and a completion time c_j, which is the date at which the processing of the job is finished. If the different jobs have priorities, a weight w_j can be associated with each job to reflect the priority. Any job may require a single resource or a set of resources.

A *resource* $i \in M$ is an equipment, an operator or a financial capital. A resource i may be renewable or non-renewable. Renewable resources are available at each period without being depleted (*e.g.* labor and many types of equipment). Non-renewable resources are depleted as they are used (*e.g.* capital and raw

materials). The resource usage may vary over the duration of the job depending on the characteristics of jobs. Some jobs, once begun, may not be interrupted until completion. Alternatively, some jobs may be interrupted at any time, possibly with some corresponding cost. Job and resource characteristics will define the constraints of the scheduling problem. Thus, constraints are often task-based and/or resource-based.

The *constraints* define the feasibility of a schedule while the *objectives* define the optimality of a schedule. Objectives should be optimized while constraints must be satisfied. The objectives often relate to performance measures called criteria. The criteria that are commonly used are related to time, resource usage, or to costs induced by the schedule.

A *feasible schedule* satisfies all of the constraints. An *optimal schedule* not only satisfies all of the constraints, but also has the best performance measure over all the feasible schedules. Schedules may be represented by Gantt charts. In scheduling, Gantt charts are often machine-oriented (Figure 2.1).

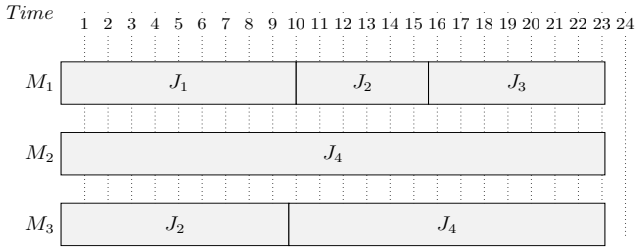


FIGURE 2.1: A machine oriented Gantt chart

2.1.2 Graham notation and problem classification under certainty

The theory of scheduling is characterized by an unlimited number of problems (see *e.g.* Błażewicz *et al.* (2012), Pinedo (2012)). To classify deterministic scheduling problems under certainty, Graham *et al.* (1979) proposed a three-field notation $\alpha|\beta|\gamma$ where α describes the machine environment, β describes the job characteristics and γ describes the optimality criteria. This classification is widely used in the literature.

Machine environment: The machine environment is characterized by the field α which is composed of two parameters α_1 and α_2. α_1 describes the machine environment, it can be $\emptyset, P, Q, R, F, J, O$.

If $\alpha_1 = \emptyset$, then all jobs are processed on a *single machine*.

If $\alpha_1 \in \{P, Q, R\}$, then we have parallel machines, *i.e.* , each job can be processed on each of the machines. If $\alpha_1 = P$, then the parallel machines are *identical parallel machines*. Thus the processing time of job j on machine i is the same for all machines $p_{ij} = p_j$. If $\alpha_1 = Q$, then the parallel machines are *uniform parallel machines*, *i.e.* $p_{ij} = p_j/v_i$ where v_i is the speed of machine i. Finally, if $\alpha_1 = R$, then the parallel machines are *unrelated parallel machines*, *i.e.* $p_{ij} = p_j/v_{ij}$ where v_{ij} is job dependent speeds of machine i. Parallel machine problems are considered as a generalization of single machine problems.

If $\alpha_1 \in \{F, J, O\}$, then we have multi-task models. Non-renewable resources are depleted as they are used (*e.g.* capital and raw materials). When $\alpha_1 = F$, we deal with a flow shop, it means that the order of execution on different machines is the same for all jobs. We determine a job shop problem by $\alpha_1 = J$, in

this configuration each job has its own order. Finally, $\alpha_1 = O$ is used for open shop problem where the order of execution on the machines is totally free.

α_2 denotes the number of machines if it is a positive integer. If $\alpha_2 = m$ then *m* is an arbitrary but fixed number. Whereas, if the field α_2 is empty, the number of machines is arbitrary.

Job characteristics The field β describes the job and machine characteristics, it often describes the constraints of the scheduling problem. β may be composed of many sub-fields, for instance:

If $\beta_1 = Split$, the jobs can be split into independent tasks, and the task can be processed with allowed parallelism. If $\beta_1 = pmtn$: the processing of a job can be interrupted and later resumed (on the same or on another machine). A job may be interrupted several times. But, the parts of the same job cannot be processed in parallel.

If $\beta_2 = prec$, then precedence constraints exist between jobs. Certain jobs must be completed before another job can be started.

If $\beta_3 = r_i$, then release date may be specified for each job.

Similarly, if $\beta_4 = d_i$, then a deadline d_i is imposed for each job. When the completion time of a job exceeds its due date, the job is late.

We notice that when there are no constraints related to a sub-field β_k, this sub-field simply does not appear in the notation.

Optimality criteria The field γ describes the optimality criteria. There exist essentially two types of cost functions (criteria), $\gamma = f_{max}$ called bottleneck objectives and $\gamma = \sum_j f_i$ called sum objectives. The most common criteria used through literature are :

- the makespan $\max_j c_j$ which is the maximal completion time over jobs, γ is denoted C_{max},
- the total flow time where $\gamma = \sum_j c_j$ and the weighted total flow time written as $\gamma = \sum_j w_j c_j$.
- and the criteria depending on due dates such as the maximal tardiness $T_{max} = max_j max\{0, c_j - dj\}$, and the maximal lateness $L_{max} = max_j\{c_j - dj\}$.

  In these cases, we write respectively, $\gamma = T_{max}$ and $\gamma = L_{max}$. The total number of late jobs is also widely considered, in this case:

$$U_j = \begin{cases} 1 & \text{if } j \text{ is late} \\ 0 & \text{otherwise} \end{cases}$$

  and $\gamma = \sum_j U_j$

Examples: To illustrate the three field notation of Graham, we present some examples.

- $1||\sum c_j$: total flow time minimization on a single machine, and $1||\sum w_j c_j$ weighted total flow time minimization on a single machine

- $F2||C_{max}$: makespan minimization in 2-machine flow shop.

- $Rm|Split|C_{max}$: makespan minimization on m unrelated parallel machines with splitting, and $Rm|pmtn|C_{max}$: makespan minimization on m unrelated parallel machines with preemptive jobs.

2.1.3 The limits of classical scheduling approach under uncertainty

The classical approach in scheduling relies on one typical instance called the nominal instance, under which the problem data are assumed to be completely known in advance (during the generation of the schedule) and to be constant (during the execution of the schedule). The schedule performance is optimized and evaluated only with respect to the nominal instance. The resulting schedule is called the predictive schedule. Unfortunately, scheduling is recognized to be a complex and dynamic activity in which job processing times can take longer or shorter than forecast, release dates and due dates may change, jobs may have to be inserted or canceled. Besides, the resources may become unavailable due to unforeseen events as breakdowns, accidents, operator absenteeism, *etc.*. Assuming a certain data and a static environment makes the classical scheduling approach inadequate in modeling and solving scheduling problems under uncertainties or disruptions. As it was emphasized in several publications, ignoring uncertainties and disruptions can lead to:

- infeasible schedules when the uncertainties or the disruptions modify the constraints of the problem, thus the computed schedule no longer respect the constraints of the real problem,
- sub-optimal schedules when the computed schedule is still feasible, but its performance is affected,
- and unstable schedules when the frequent adjustments to restore the schedule feasibility highly modify the assignment and the sequence which leads to many re-configurations and adjustment efforts. Vieira *et al.* (2003) has pointed out that these aspects are known as nervousness syndromes.

Therefore, practically speaking, computing an optimal schedule is often less important than protecting the schedule from uncertainties and disruptions. For these reasons, the issue of scheduling under uncertainty have received a lot of attention from both academic and industrial communities.

In the next section, we focus on scheduling under data uncertainty. We survey the uncertainty types and disruptions besides their representations. Then, we review the scheduling approaches under uncertainties and disruptions. Finally, we focus on the different objectives used through literature to protect a schedule from the negative effects of uncertainties and disruptions.

2.2 Scheduling under data uncertainties and disruptions

2.2.1 Uncertainties and disruptions in scheduling

Uncertainties concern job data whose exact values cannot be predetermined with accuracy. The most faced are: processing time uncertainties (*e.g.* Daniels and Kouvelis (1995*a*), Kouvelis *et al.* (2000), Petrovic *et al.* (2008), Rahmani and Heydari (2014)), due date uncertainties (*e.g.* Aissi *et al.* (2011), Kasperski and Zieliński (2016), Ebrahimi *et al.* (2014)), and release date uncertainties (Lai and Sotskov (1999), *etc.*), job weight uncertainties (*e.g.* Pereira (2016)). Disruptions are discrete unforeseen events that occur and change the structure of the ongoing schedule, they concern both resources and jobs. The disruptions that are the most faced in production environments are machine breakdowns (*e.g.* Xiong *et al.* (2013), Shi *et al.* (2014), Wang *et al.* (2015) Ahmadi *et al.* (2016)), new job arrivals (Malve and Uzsoy (2007), Yang and Geunes

(2008), Hosseini and Tavakkoli-Moghaddam (2013), Rahmani and Heydari (2014)), job cancellations (*e.g.* Abumaizar and Svestka (1997), Jain and Elmaraghy (1997), Wang and Truong (2015), Rao and Janardhana (2016)) and rush orders (*e.g.* Jain and Elmaraghy (1997)). In the scheduling community, it makes sense to distinguish disruptions from uncertainty, but for modelers, the treatment of disruptions is quite similar to the treatment of uncertainties (we will show later that disruptions are considered as aleatory uncertainties in the modeling step).

The uncertainties and disruptions in scheduling could be divided into known uncertainties and unknown one (see the classification given in chapter 1). Known uncertainties and disruptions can be measured and represented somehow during the schedule generation while unknowns are revealed progressively during the schedule execution. Based on the level of knowledge, we can classify scheduling problems into three categories: scheduling under certain data, scheduling under known uncertainties or known disruptions, and scheduling under unknowns. Known uncertainties and disruptions are often observed in static environments while unknowns are often observed in dynamic environments. Processing times, due dates and release dates, are often treated as known uncertainties. And machine breakdowns are often treated as known disruptions (see for instance Xiong *et al.* (2013), Shi *et al.* (2014), Wang *et al.* (2015)). In contrary, new job arrivals and job cancellations are often treated as unknowns (*e.g.* Zhou *et al.* (2014) and Pei *et al.* (2016) assumed dynamic job arrivals as dynamic). Nevertheless, this classification can be different depending on the context and the application. For example, processing times could be considered as unknowns in a dynamic environment where their values are revealed progressively during the schedule execution and decision-makers do not have any knowledge to represent their values prior to the execution (see for instance Chiang *et al.* (2010)).

In scheduling problems, the practical representation have more spread than the formal representations. Under aleatory uncertainty/disruption, it is usual to construct scenarios with attached probabilities in order to represent the random parameters. And when these probabilities are unknown *i.e.* under ambiguity, the discrete scenario and interval representations are widely used to model the uncertain parameters. Moreover, to describe fuzzy parameters, it is frequent to use fuzzy intervals.

2.2.2 Scheduling approaches under uncertainties and disruptions

To deal with uncertainties and disruptions in scheduling, there are basically two approaches: proactive scheduling approach and reactive scheduling approach (see Beck and Davenport (2002),Herroelen and Leus (2004*b*) Herroelen and Leus (2005), Billaut *et al.* (2008), Ouelhadj and Petrovic (2009)).

Proactive approach algorithms anticipate the uncertainties and the disruptions in order to generate solutions whose performances (or structures) are insensitive to the potential realization of data (see the illustration in Figure 2.2), while reactive approach algorithms respond once the uncertainties are revealed (or the disruptions occur) in order to build a new schedule or to adjust an existing schedule. In completely reactive scheduling, the scheduling will generate a response to the uncertainty or the disruptions in real time (see the illustration in Figure 2.3). But, the totally reactive scheduling in real time can also result in greater cost and lower performance as explained in He and Sun (2013). The proactive approach is extensively studied since it allows the anticipation of the uncertainty and the generation of baseline schedules that does not need

FIGURE 2.2: Purely proactive scheduling approach

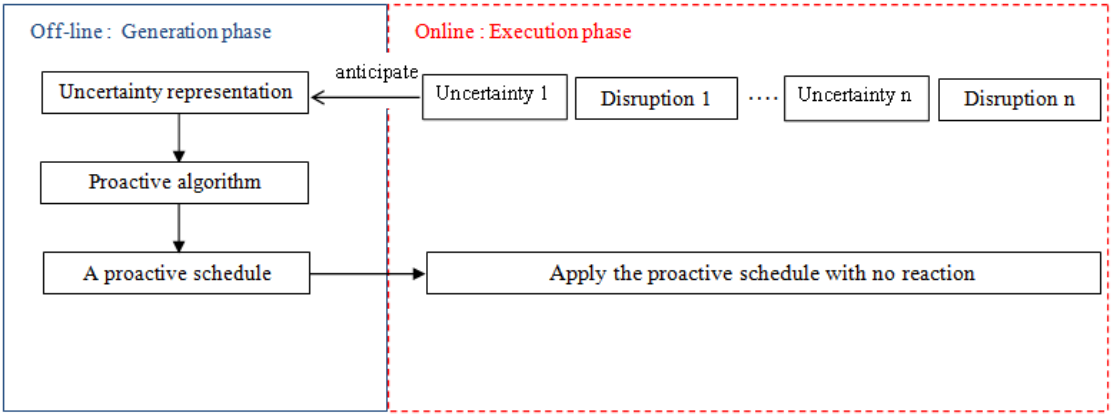
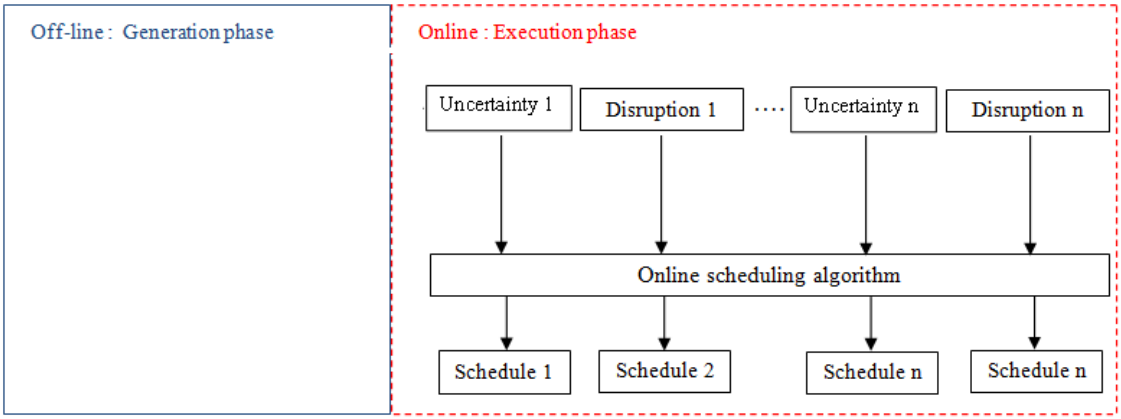


FIGURE 2.3: Purely reactive scheduling approach



to be repaired to meet the realization of uncertain data. This baseline schedule is of crucial importance for decision visibility in an uncertain environment since it gives a starting point for communication and coordination between the production, suppliers and customers. Many authors have outlined the purposes of scheduling beyond that of job-resource allocation and sequencing. Indeed, a schedule enables a better coordination in order to increase the productivity and reduce the costs. It can identify resource conflicts and ensure the availability of the required raw materials. It also identifies time periods of preventive maintenance and verifies the coordination with deliveries. A schedule is also a starting point for communication and coordination with external entities in the company's outbound (see Vieira *et al.* (2003) Aytug *et al.* (2005*a*)). The reader is referred to Mehta and Uzsoy (1998), Herroelen and Leus (2005), and Aytug *et al.* (2005*b*) for extended discussion of the proactive scheduling approach advantages. The proactive approach is suitable to deal with known uncertainties and known disruptions as explained in *Chapter 1*. Based on the uncertainty or disruption representations, we can distinguish: stochastic scheduling under probabilistic representations of uncertain data (Pinedo (2012)), robust scheduling under set representations of uncertain data (Kouvelis and Yu (1997)), and fuzzy scheduling under fuzzy representation of uncertain data (Dubois *et al.* (2003)).

According to Moukrim *et al.* (2003), an algorithm is proactive (off-line) if the schedule is determined before the execution begins.

In the reactive scheduling approach, uncertainties and disruptions are not anticipated because they are often unknowns due to the dynamic of the environment. Consequently, the schedule is built in a reactive way during the execution. Reactive scheduling is also called on-line scheduling. According to Moukrim *et al.* (2003), an algorithm is reactive (on-line) if the schedule is built during the execution.

FIGURE 2.4: Predictive-reactive scheduling approach

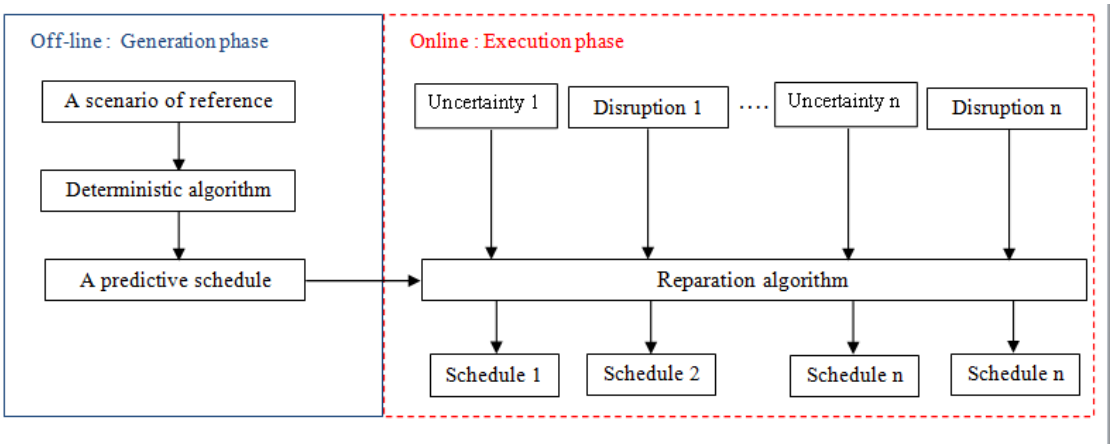
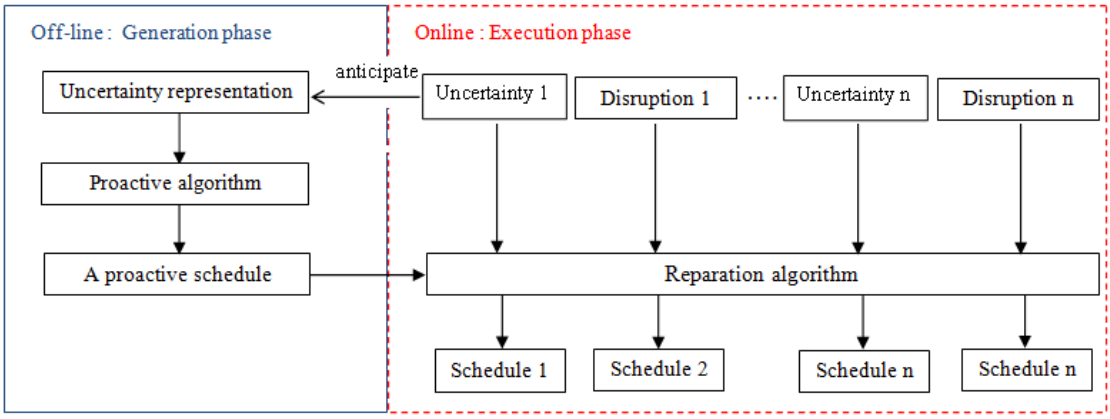


FIGURE 2.5: Proactive-reactive scheduling approach



Decision-makers consider that a baseline schedule should coordinate activities to increase the productivity and reduce the operating costs. In dynamic environments, scheduling algorithms should also react quickly to unexpected events and repair the baseline schedules. Therefore, decision-maker may opt for hybrid scheduling approaches. The hybrid approaches are necessary composed of two steps:

- Step 1: in the off-line phase, we compute a baseline schedule that will serve as a starting point of coordination between the production and the supporting activities.
- Step 2: in the on-line phase, we react to new data and disruptions and repair the baseline schedule.

Hybrid approaches could be divided into two sub-approaches: the predictive reactive approach (see the illustration in figure 2.4) and the proactive-reactive approach (see the illustration in figure 2.5). The distinction is related to the model used to compute the baseline schedule:

If the baseline schedule is computed using a deterministic algorithm that considers an instance of reference (nominal scenario), then the approach is predictive-reactive.

If the baseline schedule is computed using a proactive model (robust, stochastic, fuzzy) in which we take into account not only an instance, but a set of uncertainties, then the approach is proactive-reactive. An example of the application of such approach is given in the case of parallel machines with communication delays.

When some activities are realized, the decision-maker will have more reliable information which may be used reactively to make the necessary adjustments. The reactive strategy should use fast polynomial-time algorithms. Vieira *et al.* (2003) have defined rescheduling as the process of updating an existing schedule baseline in response to disruptions or uncertainties. Therefore, the predictive-reactive and proactive-reactive approaches can be viewed as a rescheduling process.

2.2.3 Scheduling objectives under uncertainty and disruptions: robustness and flexibility

Under uncertainties and disruptions, the decision-maker can be interested in computing a set of solutions that adapt well and that offer a freedom during the execution phase, or he/she can be interested on computing a solution that withstand a set of known uncertainties or disruptions without any change (or little changes) during the execution phase. In the former case, the objective is the flexibility of the schedule, while in the second the objective is the robustness of the schedule.

Flexibility refers to a degree of freedom available during the execution phase (Billaut *et al.* (2008)). It can be temporal (*i.e.* regarding the starting times of jobs), or sequential (*i.e.* regarding the sequence in which the jobs are sequenced), or a flexibility of assignment (*i.e.* regarding the machines that execute jobs). Given that disruptions will occur and unforeseen circumstances arise, the aim is to propose one or more solutions that adapt well to disruptions and then produce reactive decisions with fast implementation. Robustness refers to the aptitude of a solution to withstand areas of ignorance, and to provide protection against deplorable impacts, such as results that are much worse than expected Roy (2010). A robust schedule is a proactive schedule that is able to absorb some level of unexpected events without rescheduling (Herroelen and Leus (2005)). A robust schedule is often calculated by anticipating uncertainty and optimizing robustness measures that are often related to the schedule performance. There exists also a range of techniques for generating robust schedules based on the insertion of temporal protection (see slack based techniques in Gao (1996), Davenport *et al.* (2014)).

In the scheduling literature, we can distinguish two kinds of robustness: the robustness of the performance and the robustness of the structure. A robust schedule according to its performance is a schedule whose performance is insensitive to the data uncertainty and does not significantly degrade under disruptions. A robust schedule according to its structure is commonly known as stable schedule. The structure stability is commonly referred to as stability and it is defined as the aptitude of a solution to stay close to the solution of reference if small perturbations of data occur (Leus and Herroelen (2004)). A stable schedule is a

schedule whose realization does not deviate much from the schedule of reference under disruptions, and this deviation concerns the difference between the schedule of reference and the realized schedule themselves in term of structure rather than performance Sabuncuoglu and Goren (2009). A stable schedule can be obtained by optimizing stability measures. When the robustness concerns a performance, the robust schedule is usually a proactive schedule computed according to robust optimization models and it does not need to be repaired to meet the realizations of uncertain data. When robustness concerns the structure of the solution, the stability models always require the computation of a reference solution. The stability measures are relative to the job characteristics of this solution. In addition, the stability can be deployed under different approaches: proactive, predictive-reactive, proactive-reactive, and post-optimization. When the solution of reference is deterministic and the stability is optimized off-line by anticipating the uncertainties that could occur, the stability is deployed under a proactive approach. But when the solution of reference is deterministic and the stability is optimized on-line in response to an uncertainty or a disruption that occurs, the stability is then deployed under a predictive-reactive approach. Under both proactive and predictive-reactive approaches, the stability and the performance are conflicting objectives: the optimization of the stability measures leads to the degradation of the schedule performances (*e.g.* Jorge Leon *et al.* (1994); Sabuncuoglu and Goren (2009)). Therefore, most of the studies that belong to this class propose a bi-objective function to optimize the schedule stability without loss of performance. To anticipate the loss of performance, the stability could also be deployed under a proactive-reactive approach by computing a robust solution of reference and optimizing the stability to make few adjustments on-line. The stability as a post-optimization approach was introduced by Sotskov (1991). Under this approach, an optimal solution of reference is computed in a deterministic way and its stability is evaluated ex-post like in sensitivity analysis. The research question is formulated around how one can vary the data in the problem in such a way that an optimal schedule remains optimal. The answer conducts to the calculation of the stability radius of an optimal schedule which is the largest quantity of independent data variations under which an optimal schedule remains optimal.

In the next section, we will review the works dealing with uncertainties and disruptions through robustness and stability models.

2.3 Literature review on robustness in scheduling

2.3.1 Performance robustness optimization

The worst-case performance and the maximum regret are reference criteria and starting point in robustness analysis. The best worst-case performance and the best maximal regret refer respectively to min-max and min-max regret objectives that were originally defined in the strategic decision-making literature by Gupta and Rosenhead (1968); Rosenhead *et al.* (1972*b*), and that were introduced later in operations research problems by Rosenblatt and Lee (1987).

Under discrete scenario representation, Kouvelis and Yu (1997) define the worst-case performance and the maximal regret as follows:

- S is the set of discrete scenarios that represent the uncertainty,

- X is a feasible solution,
- Z^{s*} is the optimal performance criterion under scenario $s \in S$,
- $Z^s(X)$ is the performance criterion when solution X is applied to scenario $s \in S$.

The worst-case performance of solution X is given in Equation 2.1 as

$$Z_w(X) = \max_{s \in S}\{Z^s(X)\}. \tag{2.1}$$

And the maximum regret of solution X is given in equation Equation 2.2 as

$$Z_r(X) = \max_{s \in S}\{Z^s(X) - Z^{s*}\}. \tag{2.2}$$

The worst-case performance of a given solution over all the scenarios measures the maximal cost of this solution. The worst-case performance is often used for non-repetitive decisions, in environments where protection measures are indispensable. The maximum regret measures the maximal deviation from the optimal performance over all the discrete scenarios when we use a solution instead of the optimal one. The maximal regret is more suitable in situations where the magnitude of deviations varies strongly among the scenarios. In other words, minimizing the worst-case performance aims to find the cheapest schedule, while minimizing the maximal regret aims is to reduce the opportunity loss, rather than the cost.

The worst-case performance and the maximal regret measures have been used to compute robust schedules under discrete scenario representation of uncertain processing times in different machine environment studies and with various scheduling criteria.

For many polynomial solvable scheduling problems, their corresponding robust versions (min-max or min-max regret) are weakly or strongly NP-hard when taking into account the uncertainty, even in the case of the single machine (Table 2.1). For the sake of simplicity, we propose to use a short notation that combines the notations of Averbakh (2006), and Aloulou and Della Croce (2008) to represent the robust versions of a scheduling problem $\alpha|\beta|\gamma$. Let θ be the set of uncertain problem data, we will denote: *DmM*$(\alpha|\beta|\gamma;\theta)$ the min-max version of $\alpha|\beta|\gamma$ under discrete scenario representation of θ (*resp.* *DmMR*$(\alpha|\beta|\gamma;\theta)$ the min-max regret version of $\alpha|\beta|\gamma$ under discrete scenario representation of θ). Under interval representations, we denote *ImM*$(\alpha|\beta|\gamma;\theta)$ and respectively *ImMR*$(\alpha|\beta|\gamma;\theta)$.

Daniels and Kouvelis (1995*b*) considered the single machine scheduling problem with a total flow time criterion $(1||\sum c_j)$ under uncertain processing time, to formalize the concept of robust scheduling and to introduce the worst-case and the maximal regret measures under uncertain processing times. They showed that *DmM*$(1||\sum c_j; p_j)$ and *DmMR*$(1||\sum c_j; p_j)$ are NP-hard even in the case of two scenarios. Yang and Yu (2002) confirmed that minimizing the worst-case total flow time and the maximal regret in total flow time of the single machine problem under uncertain processing times are NP-hard and have proposed a dynamic algorithm, as well as polynomial heuristics to solve it. But, the complexity of these methods increases with the number of discrete scenarios. In the more general case, de Farias *et al.* (2010) proposed a cutting-plane algorithm for solving the *DmM*$(1||\sum w_j c_j; p_j)$ with the weighted sum of completion time

TABLE 2.1: Complexity of min-max scheduling problems under discrete scenario representation of data uncertainty

D(I)mM(R)$(\alpha\|\beta\|\gamma;\theta)$	*Complexities*	*References*
DmM(R)$(1\|\|\sum c_j;p_j)$	NP-hard	Daniels and Kouvelis (1995*b*), Yang and Yu (2002)
DmM$(1\|\|\sum w_jc_j;p_j)$	NP-hard	de Farias *et al.* (2010) Mastrolilli *et al.* (2013)
DmM$(1\|prec\|f_{max};p_j)$	Polynomial	Aloulou and Della Croce (2008)
DmM$(1\|prec\|f_{max};d_j)$	Polynomial	Aloulou and Della Croce (2008)
DmM$(1\|prec\|f_{max};p_j,d_j)$	Polynomial	Aloulou and Della Croce (2008)
DmM$(1\|\|\sum U_j;p_j)$	NP-hard	Aloulou and Della Croce (2008)
DmM$(1\|\sum U_j;d_j)$	Polynomial	Aloulou and Della Croce (2008)
DmM$(1\|\|\sum U_j;d_j)$	NP-Hard	Aissi *et al.* (2011)
DmM$(1\|\|\sum U_j;p_j,d_j)$	NP-hard	Aloulou and Della Croce (2008)
DmMR$(F2\|\|C_{max};p_j)$	NP-hard	Kouvelis *et al.* (2000)
DmMR$(F2\|\|C_{max};p_j)$	strongly NP-hard	Kasperski *et al.* (2012)
IMm$(1\|prec\|L_{max};p_j,d_j)$	Polynomial	Kasperski (2005)
ImMR$(1\|prec\|\sum c_j;p_j)$	NP-hard	Kasperski and Zielienski (2008)
ImMR$(1\|prec\|\sum c_j;p_j)$	NP-hard	Conde (2010)
ImMR$(1\|\|\sum w_jc_j;p_j)$	NP-hard	Pereira (2016)
ImMR$(1\|\|\sum w_jU_j;p_j)$	NP-hard	Drwal (2017)
ImMR$(Pm\|\|C_{max};p_j)$	NP-hard	Xu *et al.* (2013)
ImMR$(Pm\|\|\sum c_j;p_j)$	NP-hard	Drwal and Rischke (2016)
ImMR$(Qm\|\|\sum c_j;p_j)$	NP-hard	Xu *et al.* (2014)
ImMR$(Rm\|\|\sum c_j;p_j)$	NP-hard	Conde (2014)
ImMR$(Rm\|\|\sum c_j;p_j)$	NP-hard	Siepak and Józefczyk (2014)

criterion. Several inapproximability results for this problem are established in Mastrolilli *et al.* (2013). Aloulou and Della Croce (2008) studied several robust single machine scheduling problems. They proved that the minimization of the worst-case of general maximum cost function $f_{max} \in \{C_{max}, L_{max}, T_{max}\}$ with job precedence under uncertain processing times (*i.e.* *DmM*$(1|prec|f_{max};p_j)$) is optimally solved in a polynomial time by the MinMax-Lawler algorithm. Moreover, they proved that *DmM*$(1||\sum U_j;p_j)$ is NP-hard even if there are two scenarios. Kouvelis *et al.* (2000) considered the two machines permutation flow shop problem with makespan minimization under uncertain processing times. $F2||C_{max}$ is is known to be polynomially solvable by using Johnson's algorithm. Kouvelis *et al.* (2000) showed that the minimization of the maximal regret in makespan of two machine permutation flow shop under uncertain processing times, *DmMR*$(F2||C_{max};p_j)$, is weakly-NP-hard. For the same scheduling problem, Kasperski *et al.* (2012) has shown later that *DmM*$(F2||C_{max};p_j)$ and the *DmMR*$(F2||C_{max};p_j)$ are strongly NP-hard even for two scenarios.

Other uncertainties were considered in scheduling problems as due dates (d_j), release dates (r_j) and job priorities (w_j). Aloulou and Della Croce (2008) proved that *DmM*$(1|prec|f_{max};d_j)$ can be optimally solved in polynomial time by means of Lawler's algorithm applied to the worst-case artificial scenario under which all the processing times take their maximal values. They proved that *DmM*$(1|prec|f_{max};d_j,pj)$ is

solved to optimality in polynomial time by the MinMax-Lawler algorithm. Besides, they showed that the minimization of the number of late jobs on a single machine when the processing times are certain and due dates are uncertain, *DmM*$(1||\sum U_j; d_j)$, can be optimally solved by means of Moore's modified algorithm applied to the worst-case artificial scenario s_{max} under two due date scenarios. For a variable number of scenarios and two distinct due dates over all scenarios, the problem is NP-hard in the strong sense and non-approximable in pseudo-polynomial time with approximation ratio less than 2 according to Aissi *et al.* (2011). It is polynomially solvable if the number of scenarios and the number of distinct due dates over all scenarios are given constants. When the job weights are uncertain, Aloulou and Della Croce (2008) proved that the minimization of the total weighted flow time in single machine *DmM*$(1||\sum w_j c_j; w_j)$ is NP-hard even if the number of scenarios equals 2 and the processing time of each job equals the unit. The minimization of the weighted sum of late jobs under uncertain weights is known to be NP-hard for two weight scenarios, strongly NP-hard and hard to approximate within any constant factor if the number of weighted scenarios is a part of the input (The reader is referred to Kasperski and Zieliński (2014)).

When we consider the interval representation of uncertain parameters as an extension of discrete scenario representation where the number of scenarios is infinite, we could also focus on the following studies. Kasperski (2005) have considered the single machine sequencing problem with maximum lateness criterion where the parameters are uncertain and belong to intervals. He has shown that the *IMm*$(1|prec|L_{max}; p_j, d_j)$ is polynomial. Lebedev and Averbakh (2006) have proved that the min-max regret version of under uncertain processing times that belong to intervals, *i.e.* *ImM*$((1||\sum c_j; p_j)$ is NP-hard. They showed that in the case where all intervals of uncertainty have the same center, the problem can be solved in polynomial time if the number of jobs is even, and is NP-hard if the number of jobs is odd. Kasperski and Zielienski (2008) considered the general *ImMR*$(1|prec|\sum c_j; p_j)$, which is total flow time minimization with precedence constraints, under uncertain processing times. The general deterministic problem is strongly NP-hard (Lenstra and Rinnooy Kan (1978)). Kasperski and Zielienski (2008) proved that the optimal schedule under the midpoint scenario is a 2-approximation for *ImMR*$(1|prec|\sum c_j; p_j)$. Conde (2010) developed a 2-approximation method for min-max regret optimization problems which extends the work of Kasperski and Zielienski (2008) from finite to compact constraint sets. Pereira (2016) considered the single machine sequencing problem with total weighted flow time under interval processing times. He showed that*ImMR*$(1||\sum w_j c_j; p_j)$ is NP-hard and presented an exact branch-and-bound method to solve the problem. Conde (2014) has proposed a Mixed Integer Programming formulation of the unrelated parallel machine with total flow criterion under uncertain processing times. Each processing time belongs to a known interval. In this context, they showed that the problem is NP-hard and considered that an optimal solution of *ImMR*$(Rm/sumc_j; p_j)$ is a suitable approximation to the optimal schedule under an arbitrary choice of the possible processing times. Siepak and Józefczyk (2014) also the same problem. They proposed a simple 2-approximate middle interval time efficient algorithm besides a scatter search based heuristic algorithm. They showed via experiments that this latter is more time consuming, but it is better in terms of the quality of solutions. Xu *et al.* (2013) addressed the min-max regret version of the makespan minimization on identical parallel machines under interval processing times. They showed that *ImMR*$(Pm||C_{max}; p_j)$ is NP-hard and they proved that a regret-maximizing scenario for any schedule belongs to a finite set of extreme point scenarios. Then they derived

two exact algorithms using a general iterative relaxation procedure. Xu *et al.* (2014) considered the total flow time minimization on uniform parallel machines under interval processing times. They proved that the optimal schedule under the midpoint scenario is a 2-approximation for the *ImMR*$(Qm||\sum c_j; p_j)$. Drwal and Rischke (2016) showed that the *ImMR*$(Pm||\sum c_j, p_j)$ is strongly NP-hard. Cwik and Józefczyk (2015) considered the makespsan minimization in a flow-shop problem with interval processing times. The maximal regret is used to evaluate a solution which gives the min-max regret binary optimization problem. They proposed an evolutionary heuristic solution algorithm to solve *ImMR*$(F||C_{max}; p_j)$ and compared it with a simple middle interval heuristic algorithm for three machines instances. Drwal (2017) considered the minimization of the total weight of late jobs in a single machine under interval processing times. They showed that *ImMR*$(1||\sum w_j U_j; p_j)$ is NP-hard and proposed a heuristic algorithm based on mixed-integer linear programming to compute a robust solution.

Besides theoretical scheduling, the robustness under scenario uncertainty or interval uncertainty has been spread to several applications as project scheduling, line balancing, energy optimization, *etc.*. In project scheduling, Artigues *et al.* (2013) examined the min-max regret Resource-Constrained Project Scheduling Problem. They showed that the robust problem turned out to be exceptionally difficult, in that even exact objective-function evaluation is intractable and computationally overly demanding, even for medium-sized instances. They implemented a scenario-relaxation algorithm that produces optimal solutions but requires excessive running times even for medium-sized instances. Furthermore, they proposed a scenario-relaxation-based heuristic that produces high-quality solution for medium-sized instances in less CPU times. In robust line balancing, Dolgui and Kovalev (2012) studied the line balancing problem under uncertain processing times where the objective is to minimize the worst-case maximum execution time of the same station. They represented uncertainty by a set discrete scenarios and proposed an approach to reduce its cardinality. They showed that several special cases of the problem are NP-hard and strongly NP-hard. Furthermore they suggested enumerative dynamic programming algorithms and problem-specific polynomial time algorithms for some cases. For more examples, the reader is referred to Bentaha *et al.* (2015). The min-max and min-max regret criteria lead to a risk-averse decision since they assume that decision makers risk-averse (or pessimistic). It follows from the fact, that the computation of schedules that minimize the worst-case performance (*resp.* the maximal regret) is particularly constrained with the worst-case scenarios, ignoring the information connected with the remaining scenarios. In some context where protection measures should be taken, the choice of these criteria is crucial. But, in context where decision makers are less risk averse, other approaches can be applied. Instead of seeking for a solution that optimize the robustness measure, the decision makers impose conditions that solutions must satisfy in order to be considered as robust. For instance, Daniels and Carrillo (1997) defined a β-robustness measure that is based on the likelihood of achieving system performance no worse than a given target level. Daskin *et al.* (1997) proposed a α-reliable min-max regret model to compute a solution that minimizes the maximum regret with respect to a selected subset of scenarios whose probability of occurrence is at least α (a value defined by the decision maker). Kalaï and Lamboray (2007), Kalaï *et al.* (2010) proposed an approach called lexicographic α-robustness which considers all scenarios in lexicographic order from the worst to the best, and where α is a tolerance threshold in order to not discriminate among solutions with similar values.

The expected performance is known to be the most-appropriate for a risk-neutral decision maker. It is widely used in scheduling under both random processing times and under random machine breakdowns Jorge Leon *et al.* (1994); Wu *et al.* (1999); Goren and Sabuncuoglu (2009)).The study of the performance distribution is also important when dealing with stochastic scenarios. For example, in project scheduling under stochastic durations, Ballestin and Leus (2009) minimized the expected makespan and studied the distribution of makespan realizations for a given scheduling policy. The analytic calculation of the expected performance measure is often complicated when the probability distribution is continuous which justify the use of simulation or surrogate measures to estimate this value. When decision-makers are able to assign discrete probabilities or sampling, the expected performance of solution X could be calculated as given in Equation 2.3

$$Z_e(X) = \sum_{s \in S} \{p^s Z^s(X)\}, \tag{2.3}$$

where p^s is the probability attached to scenario s.

The expected performance minimization is often combined with a stability objective as we will see in Table 2.2. In the next section, we focus on the studies that considers the stability issue in scheduling under uncertainty.

2.3.2 Structure robustness optimization

To construct stable schedules, the stability measures proposed in the literature are basically related to: the starting times or the completion times deviations (*e.g.* Wu *et al.* (1993); Abumaizar and Svestka (1997); Mehta (1999); Dong and Jang (2012)), the sequence deviation (*e.g.* Wu *et al.* (1993); Abumaizar and Svestka (1997); Cowling and Johansson (2002); Leus and Herroelen (2004)), the number of disrupted jobs (*e.g.* Ozlen and Azizoğlu (2011)).

Starting time and completion time deviations are very useful measures of stability, especially in shop environments where secondary resources such as tooling and handling should be prepared. A change in job starting or completion time may incur carrying costs if the material is delivered earlier than expected, or perhaps more importantly, rush order costs if the tools and material are requested earlier than expected (Wu *et al.* (1993)). An example of such stability measure in Equation 2.4 is given by summing the absolute value of the differences in job starting times between the realized schedule and the schedule of reference:

$$\text{Starting time deviation} = \sum_{j \in J} \mid s_j - s_j^{initial} \mid \tag{2.4}$$

Where s_j and $s_j^{initial}$ are respectively the starting time of job j in the realized schedule and the initial schedule. The absolute values permit to report both delay and rush.

The sequence deviation measure is critical if machine setups are prepared in advance based on the initial sequence. For instance, jobs may wait in a sequence queue, and tooling may be planned in advance according to the initial sequence. Thus, a sequence deviation will incur costs in handling and reallocating the jobs, and

re-planning the tools changeover (Wu *et al.* (1993)). An example of such measure is defined by Abumaizar and Svestka (1997) as follows:

- S_1: set of jobs processed before operation j in the initial schedule.
- S_2: set of operations processed after operation j in the new schedule.
- N_{ij}: cardinality of $S = S_1 \cap S_2$.

The formula in Equation 2.5 gives the measure of jobs sequence deviation over all the machines:

$$\text{sequence deviation} = \sum_{i \in I} \sum_{j \in J} N_{ij} \tag{2.5}$$

Starting/Completion times deviations, and sequence deviation are widely used in single machine, job shop and project scheduling problems under stochastic machine breakdowns and/or stochastic processing times. The vast majority of these studies are bi-objective. They consider both stability and efficiency, for instances:

Wu *et al.* (1993) considered the makespan minimization on a single machine under stochastic machine breakdowns. They proposed a bi-objective heuristics in order to optimize the stability and to maintain a good makespan. For this purpose, they used the total deviation in starting times and the sequence deviation as stability measures. Their computational results show that the schedule stability can be increased significantly with little or no sacrifice on the makespan. Abumaizar and Svestka (1997) considered a job shop problem affected by disruptions modelled as stochastic machine breakdowns. They also considered starting time deviation and sequence deviation as stability measures, and they presented a heuristic that optimizes both the makespan and the stability measures to produce an efficient and stable schedule.

Mehta and Uzsoy (1998) consider job shop environment under machine breakdowns. They present a linear programming based heuristic that minimizes the deviation of the starting times compared with a schedule that contains ample slack, while respecting a deadline for the project. They insert additional idle time into the schedule to absorb the impact of breakdowns, and invoke earliness or lateness penalties whenever the last operation of a job ends sooner or later than planned. Mehta (1999) considered the maximum lateness minimization on a single machine under stochastic machine breakdowns. The effects of disruption on planned activities are measured through job completion time deviation. Cowling and Johansson (2002) considered the single machine problem with the objective of minimizing the average completion time with no allowed preemption. They used the sum over all jobs of the starting time and completion time deviation divided by the number of jobs to measure the impact of moving from the initial schedule to the repaired schedule. Mehta and Uzsoy (1998) considered job shop environment under stochastic machine breakdowns. They presented a linear programming based heuristic that minimizes the deviation of the starting times, while respecting a deadline for the project. They inserted additional idle time into the schedule to absorb the impact of breakdowns, and invoked earliness or lateness penalties whenever the last operation of a job ends sooner or later than planned. Herroelen and Leus (2004*a*) adapted the linear programming based heuristic that was developed by Mehta and Uzsoy (1998) to run in a project scheduling environment. They gave two

mathematical programming models to minimize the expected weighted starting time deviation in order to construct a stable baseline project schedule, when the activities durations increase. Yang (2013) considered a single machine rescheduling problem. Disruptions such as new job arrivals and cancelled jobs occur after the initial scheduling. The objective is the minimization of the completion times deviation while preserving a small makespan. The artificial due dates for the remaining jobs are set to completion times in the original schedule while newly arrived jobs do not have due dates. The objective of rescheduling is to minimize the maximum earliness without tardiness. They developed three simple heuristics and demonstrated that two heuristics perform much better than the other one. Yin *et al.* (2016) considered identical parallel machines subject to machine disruptions with the objective of minimizing the total completion time. The objective is to schedule the affected jobs and maximize the stability with respect to the original schedule. Schedule stability is measured by the completion time deviation. The study is bi-objective and focuses on the trade-off between the total completion time of the adjusted schedule and completion time deviation. They developed pseudo-polynomial-time solution algorithms for the problem with a fixed number of machines and conduct extensive numerical studies to evaluate the performance of the proposed algorithms.

In project scheduling, Herroelen and Leus (2004*a*) devoted their attention to the development of the stable pre-schedule under uncertainty. They proposed two mathematical formulations to minimizing the expected weighted deviation of starting times under job disruption. In the former, they anticipate the disruption of one job duration while in the second one, they anticipate two disruptions. To serve as benchmark, they proposed three additional models. Some of these models are adapted from original version provided by the literature of machine scheduling as the linear programming based heuristic given in Mehta and Uzsoy (1998).

Only few studies considered both stability and robustness. Jorge Leon *et al.* (1994) considered the job shop problem with stochastic machine breakdowns and processing time variations. The authors addressed the case of single disruption which serves as a basis for treatment of the more general case. The model is bi-objective and is based on the minimization of the expected makespan of the realized schedule (robustness) and the expected deviation from the original schedule makespan (stability). The experimental results showed that the bi-criterion approach significantly outperforms the deterministic approach based only on makespan. Goren and Sabuncuoglu (2008) addressed the problem of finding robust and stable schedule in a single machine environment under stochastic machine breakdowns. Many schedule performances were studied: makespan, total flow time, and total tardiness. The robustness of the schedule is measured through the expected performance of the realized schedule, while the stability is measured through the expected sum of absolute deviations in job completion times. The surrogate measures of robustness and stability are embedded in a tabu-search algorithm. Goren and Sabuncuoglu (2009) extended the list of stability measures to the sum of the squared differences of the job completion times, the sum of absolute differences of the job completion times and the sum of the variances of the realized completion times.

In project scheduling, Van de Vonder *et al.* (2006) addressed the trade-off between stability and robustness in resource constrained project scheduling. They proposed a heuristic algorithm that minimizes the weighted deviation of starting times and provided an extensive simulation experiment to assess the trade-off between stability and robustness (measured through the completion time).

In parallel machine scheduling, the authors used, particularly, two measures which are the number of disrupted jobs and the total cost of reassignments. As the jobs could be assigned to different machines after the disruption, the research question is often about finding the solution that offers the best trade-off between stability and performance. Azizoglu and Alagöz (2005) defined the number of disturbed jobs as given in Equation 2.6 such that

$$\text{Disturbed jobs} = \sum_j n_j \text{ where } n_j = \begin{cases} 1 & \text{if } j \text{ changes of machines in the new schedule} \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

Azizoglu and Alagöz (2005) considered a rescheduling problem on identical parallel machine environment subject to stochastic machine breakdowns. They addressed the trade-off between between efficiency and stability. They considered the total flow time as an efficiency measure while the stability is measured in terms of the number of disrupted jobs that are processed on different machines in the original and new schedules. Özlen and Azizoğlu (2009) considered unrelated parallel machine subject to stochastic machine breakdowns. They consider the total flow time as a performance measure and the total reassignment cost caused by the schedule deviation as stability measure. They illustrated that the efficient schedules with respect to the two objectives can be found in polynomial time. Ozlen and Azizoğlu (2011) also considered unrelated parallel machines under stochastic machine breakdowns. The affected jobs are rescheduled by optimizing the total flow time, and the total reassignments cost caused by the differences between the initial and current schedules. They provided polynomial-time solution methods to the problems of minimizing total disruption cost among the minimum total flow time schedules and minimizing total flow time among the minimum total disruption cost schedules. Besides, they proposed exponential-time algorithms to generate all efficient solutions. The computational tests on large size problem instances show that the algorithm finds the best solution by generating only a small portion of all efficient solutions.

Curry and Peters (2005) examined the rescheduling in parallel machine problems due to the arrival of new jobs. They considered machine reassignment costs to minimize the instability. Their simulation experiments showed significant gains in schedule stability by selecting the alternative optimal solution with the fewest machine reassignment cost. Besides, Kaplan and Rabadi (2015) considered the parallel machine disrupted by the arrival of new jobs, the departure of an existing job, and changes to job priority. A bi-criteria objective function was considered to simultaneously minimize both the total weighted tardiness and schedule instability. They formulated the problem by extending the mixed integer linear programming model of the scheduling problem in Kaplan and Rabadi (2012). Five heuristic algorithms are introduced to handle the bi-objective problem using algorithms by Kaplan and Rabadi (2013). Their experiments solved small size and large size problems for different types of disruptions.

The complexity of machine scheduling for stability is rarely addressed in the literature, to the best of our knowledge, only Leus and Herroelen (2005) studied the complexity of machine scheduling for stability. They used the expected weighted deviation in start times as stability measure. They showed that single machine, single machine with unequal ready times, and single machine with precedence constraints under single-disruption are NP-hard. Then, they considered parallel machine problem. They also showed that the

single-disruption stability problem with free number of parallel machines is strongly NP-hard.

To evaluate the stability of optimal solutions, the stability as post-optimality analysis has been used in single machine scheduling (Sotskov and Lai (2012), in job (general) shop scheduling problems (Sotskov (1991); Sotskov *et al.* (1997)), in assembly line problem (Sotskov and Dolgui (2001); Sotskov *et al.* (2006). Sotskov *et al.* (2010) discussed the application of this approach to the job (and general) shop problem with the makespan and the total flow time criteria. This approach is based on the separation of the structural input data (precedence and capacity constraints) from the numerical input data. The approach is based on an improved stability analysis of an optimal digraph. In application to semi-conductor production environment, Rossi (2003, 2010) presented an approach for measuring the stability of a configuration of parallel multi-purpose machines under demand uncertainty. They assessed the minimum magnitude of forecast demand perturbations that may lead to breaking the deadline provided by the decision maker. They showed accordingly the practical use of "the stability radius" in industry. The stability analysis approach was also applied to several scheduling problems, where the deterministic versions allow a polynomial-time solution. Sotskov *et al.* (2010) presented additional results for the two-machine flow (and job) shop problem with interval processing times both the off-line scheduling problem and the on-line problem is considered. The stability approach was also implemented for the single machine mean flow time problem with interval processing times (Sotskov *et al.* (2010, 2011)).

2.4 Literature review synthesis and research potential issues

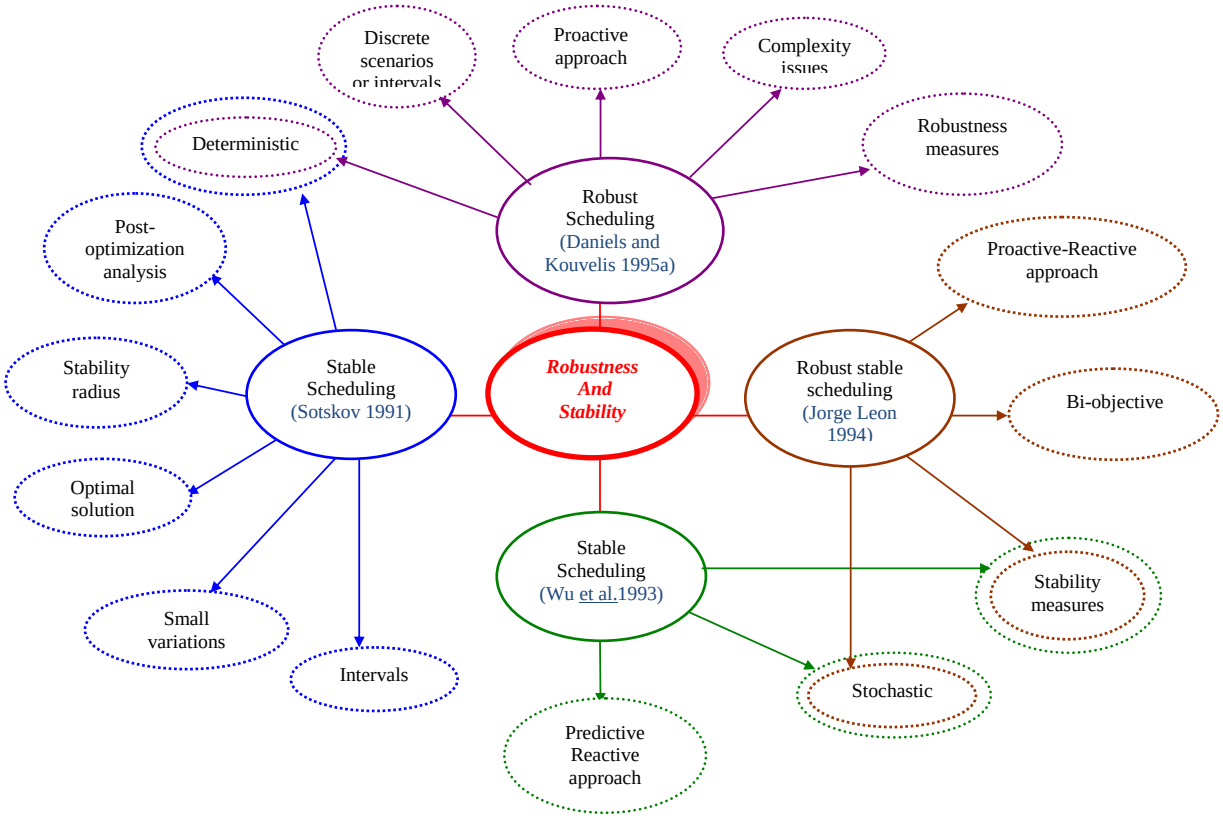
2.4.1 Literature synthesis

We have surveyed published works on scheduling under uncertainties for robustness and/ or stability objective published between 1995 and 2017 (Table 2.2). The total population of published works on the topic falls into one of four clusters organized around 4 seminal publications:

- Cluster 1 (Robust scheduling) includes all the works based on the approach of Daniels and Kouvelis (1995*a*). These works belong to the purely proactive approach aim to compute robust schedule under ambiguity. Most of the addressed problems belong to theoretical scheduling problems. Single machine environment is the most considered. Processing times, followed by due dates and then job weights are the most frequently analysed uncertainties. The uncertain data are modeled by discrete scenarios (or intervals) and the authors use min-max (regret) objectives or extension of these objectives. An unavoidable question under these works is related to the complexity of the robust versions of the scheduling problems. The vast majority of these robust version is NP-hard. Therefore, the authors propose, design and evaluate multiple algorithms to compute the robust solutions.

- Cluster 2 (Stable scheduling) includes all the works that extend the research question of Wu *et al.* (1993). These works which belong to the predictive reactive approach use bi-objective function to optimize the stability and to maintain a good performance under stochastic uncertainty or disruptions. The authors use the expected performance as performance objective and stability measures.

FIGURE 2.6: A simplified clustering diagram on robustness and stability issues in scheduling



- Cluster 3 (Robust Stable scheduling) includes all the works that extend the research question of Jorge Leon *et al.* (1994). These works which belong to the proactive-reactive approach or to the purely proactive approach. They use bi objective function to optimize the robustness and the stability of the schedule under stochastic uncertainty/disruptions, the processing times followed by machine breakdowns being the most frequently addressed uncertain parameters.

- Cluster 4 (Stability as post-analysis) includes all the works that extend Sotskov (1991) research questions. Under this approach, an optimal solution of reference is computed in a deterministic way and its stability is evaluated a-posteriori under small perturbations of the data. The research question is formulated around how one can vary the data in the problem in such a way that an optimal schedule remains optimal. The answer conducts to the calculation of the stability radius (*resp.* a ball or a box).

TABLE 2.2: Literature review synthesis (an overview)

Approach	Scheduling problem		Context and			Objectives		References
	Environment	Performance	data	event	Model	Rob	Stab	
Proactive	Single machine	Makespan	p_j		D	x		Aloulou and Della Croce (2008)
	Single machine	Total flow time	p_j		D	x		Daniels and Kouvelis (1995*a*)
	Single machine	Total flow time	p_j		D	x		Yang and Yu (2002)
	Single machine	Total flow time	p_j		D	x		Lebedev and Averbakh (2006)
	Single machine	Total flow time	p_j		D	x		Kasperski and Zielienski (2008)
	Single machine	Total flow time	p_j		D	x		Conde (2010)
	Single machine	Total weigted flow time	w_j		D	x		Aloulou and Della Croce (2008)
	Single machine	Total weigted flow time	p_j		D	x		de Farias *et al.* (2010)
	Single machine	Total weigted flow time	p_j, w_j		D	x		Mastrolilli *et al.* (2013)
	Single machine	Total weigted flow time	p_j, w_j		D	x		Pereira (2016)
	Single machine	Maximum lateness	p_j, d_j		D	x		Kasperski (2005)
	Single machine	Maximum latetness	d_j		D	x		Aloulou and Della Croce (2008)
	Single machine	Number of late jobs	d_j		D	x		Aloulou and Della Croce (2008)
	Single machine	Number of late jobs	d_j		D	x		Aissi *et al.* (2011)
	Single machine	Number of weighted late jobs	p_j		D	x		Drwal (2017)
	Flow shop	Makespan	p_j		D	x		Kouvelis *et al.* (2000)
	Flow shop	Makespan	p_j		D	x		Kasperski *et al.* (2012)
	Parallel machines P	Makespan	p_j		D	x		Xu *et al.* (2013)
	Parallel machines P	Total flow time	p_j		D	x		Drwal and Rischke (2016)
	Parallel machines Q	Total flow time	p_j		D	x		Xu *et al.* (2014)
	Parallel machines R	Total flow time	p_j		D	x		Conde (2014)
	Parallel machines R	Total flow time	p_j		D	x		Siepak and Józefczyk (2014)
	Job shop	Makespan	p_j	$brdwn$	S	x	x	Jorge Leon *et al.* (1994)
	Single machine	Makespan		$brdwn$	S	x	x	Goren and Sabuncuoglu (2008)
	Single machine	Total flow time		$brdwn$	S	x	x	Goren and Sabuncuoglu (2008)
	Single machine	Total tardiness		$brdwn$	S	x	x	Goren and Sabuncuoglu (2008)
	Single machine	Total flow time	p_j	$brdwn$	S	x	x	Goren and Sabuncuoglu (2009)
	Single machine	Total tardiness	p_j	$brdwn$	S	x	x	Goren and Sabuncuoglu (2009)
	Flexible Job shop	Makespan		$brdwn$	S	x	x	Al-Hinai and ElMekkawy (2011)
	Job shop	Makespan		$newj$	S	x	x	Rahmani and Heydari (2014)
	Single machine	Makespan		$brdwn$	S		x	Wu *et al.* (1993)
	Job shop	Makespan		$brdwn$	S		x	Jorge Leon *et al.* (1994)
	Project	Makespan		$brdwn$	S		x	Mehta (1999)
	Project	Makespan	p_j		S		x	Herroelen and Leus (2004*a*)
Predictive Reactive	Job shop	Makespan		$brdwn$	S		x	Abumaizar and Svestka (1997)
	Parallel machines P	Total flow time		$brdwn$	S		x	Azizoglu and Alagöz (2005)
	Parallel machines P	Total flow time		$brdwn$	S		x	Özlen and Azizoğlu (2009)
	Parallel machines R	Total flow time		$brdwn$	S		x	Özlen and Azizoğlu (2009)
	Flow shop	Makespan		$brdwn$	S		x	Katragjini *et al.* (2013)
	Project	Makespan	p_j		S		x	Van de Vonder *et al.* (2006)
	Project	Makespan	p_j		S		x	Herroelen and Leus (2004*a*)
Post-Optim	Job shop	Makespan	p_j		S		x	Sotskov (1991)
	Job shop	Total flow time	p_j		S		x	Sotskov *et al.* (2010)
	single machine	Flow time	p_j		S		x	Sotskov and Lai (2012)

2.4.2 Research potential issues

From these clusters, we can remark that there exist a lot of opportunities for future works. Here some points:

i) Cluster 1 works can provide schedules that are immune against uncertainties and disruptions for a described uncertainty. But, some robust solutions may fail when the future turns out differently than considered. A robust model may not guarantee a good performance under a scenario outside the set of considered discrete scenarios or interval. The robustness oriented studies rarely addressed these questions. There is a lack of studies that address the stability of robust solutions from both structural and performance viewpoints.

ii) Cluster 2 and Cluster 3 works deal basically with stochastic uncertainties and disruption (frequently processing times and machine breakdowns). The future research might focus on other uncertainties and disruptions as the due dates, the release dates, the setup times, *etc.*.They can also model the uncertainties and the disruptions using other representations as intervals or scenarios instead of probability distributions. The existing works in cluster 2 can also be extended to consider robustness objective besides the stability objective. The results provided by the works of Cluster 1 can be embedded in these works to provide robust solutions using min-max and min-max regret objectives.

iii) Stability as post optimization analysis is a promising approach for further research since it focus until now on deterministic solutions and use complicated measure to answer only one question "what is the data change for which the optimality is not affected?". This question is important in Cluster 1 as we have pointed out. Thus, the basic idea of stability analysis can be used to answer the question. But we can also address the stability with a different research question as What is the impact of a given data change on the robust solution?

TABLE 2.3: Parallel machine scheduling under uncertainty: review synthesis

Scheduling problem				Objectives			References
Environment	Performance	Uncertainty	disruption	Model	Rob	Stab	
Pm	Makespan	p_j		D	x		Xu *et al.* (2013)
Pm	Total flow time	p_j		D	x		Drwal and Rischke (2016)
Qm	Total flow time	p_j		D	x		Xu *et al.* (2014)
Rm	Total flow time	p_j		D	x		Conde (2014)
Rm	Total flow time	p_j		D	x		Siepak and Józefczyk (2014)
Pm	Total flow time		$brdwn$	S		x	Azizoglu and Alagöz (2005)
Pm	Total flow time		$brdwn$	S		x	Özlen and Azizoğlu (2009)
Rm	Total flow time		$brdwn$	S		x	Özlen and Azizoğlu (2009)
Pm	Total tardiness		$new\ job$	S		x	Kaplan and Rabadi (2015)
Pm	Total tardiness		$new\ priority$	S		x	Kaplan and Rabadi (2015)
Pm	Total flow time		$brdwn$	S	x		Yin *et al.* (2016)

vi) Parallel machine scheduling problem under uncertainty, which constitutes our subject of study, is treated with different approaches in the literature for robustness or stability objective. Under robustness, the objective is to deal with uncertain processing times while under stability the objective is principally to deal with machine breakdowns. The robustness and the stability in parallel machine scheduling are rarely addressed jointly. Besides, the preemptive version is rarely addressed.

Conclusion

In this chapter, we presented the general form of the scheduling problem and define its fundamental notions. We discussed the limitations of scheduling under the hypothesis of certainty based on previous work claims. Then, we reviewed the representations of the uncertainties and disruptions in scheduling. We classified the scheduling approaches and objectives under uncertainties and disruptions. In the review, we focused on the works that are oriented toward robustness and stability.

The survey revealed that the contributions are organized around four clusters. The analysis of these clusters showed that there exist a lot of opportunities for future developments. Robust optimization can provide schedules that are immune against uncertainties and disruptions but, robust solutions may fail when the future turns out differently than considered. The robustness oriented studies do not consider this issue while stability models do. However, stability and robustness are mentioned to be conflicting objectives. Besides, most of the studies that consider robustness focus on the performance robustness and do not address the robustness of the structure that is in part very important for practitioners. Stability analysis approach focuses on deterministic solutions and use complicated measures. Based on these remarks, we will consider a parallel machine scheduling environment under processing times uncertainty under which we will address all these points:

Research question 1: How to provide robust solutions that withstand the uncertain processing times?

Research question 2: What is the robustness cost when we use a robust solution in order to withstand the uncertainty?

Research question 3: Which solutions are robust stable when confronted to a new scenario?

Chapter 3

Toward a robust schedule on unrelated parallel machines under uncertain processing times

Abstract: The goal of this chapter is to address the makespan minimization on parallel machines with job splitting (*resp.* preemption) under uncertain processing times. We show that when the processing times are uncertain, the optimal solutions computed based on the nominal scenario can lead to infeasible schedules, or schedules with poor performance. To remedy this, we anticipate the uncertainty by constructing proactively robust schedules. In the first approach, we enforce the feasibility of the schedule under the set of potential scenarios by tolerating the violations of some processing requirements. We use a weighted function of slack variables to control the violations. We show that the schedule constructed under this approach can lead to important overproduction and underproduction for some jobs. In the second approach, we guaranty the feasibility under a set of potential scenarios without violations: we construct a set of feasible solutions based on artificial scenario schedules and we choose over this set the solution that leads to the best performance in the worst-case. This second approach allows to respect the constraints of the problem under any potential scenario, and allows to construct a schedule with a good performance.

Contents

Introduction

Parallel machine scheduling was and is still a rich and promising field of research with numerous applications in manufacturing lines, computer processing, preemptive multitasking, and multi-stage systems (see Cheng and Sin (1990)). The following examples illustrate the role of parallel machines in different applications(*e.g.* semi-conductors: Rossi (2003, 2010), Bilyk and Mönch (2012), Aubry *et al.* (2012), processors: Guinand *et al.* (2004), textile: Silva and Magalhaes (2006), aircraft planning: Hancerliogullari *et al.* (2013), *etc.*).

The classical problem of scheduling parallel machines consists in sequencing n jobs on m same function machines in order to optimize the objective. Each job j must be performed on a machine with a fixed processing time p_{ij}. p_{ij} represents the processing time of job j in machine i. In parallel machine scheduling, we distinguish three environments of the machines. In the case of parallel identical machines ($\alpha = Pm$), the job processing time is independent of the machine where it is processed. For uniform parallel machines $\alpha = Qm$, each machine has a different speed. The processing time is equal to the processing requirement divided by the speed of the machine. In unrelated parallel machines $\alpha = Rm$, no particular relationship exists between the processing times on the different machines, *i.e.* there is no proportionality between the processing time of a job on a given machine and the processing time of the same job on another machine.

The literature on scheduling parallel machine problems is abundant, a deep review is provided in (Mokotoff (2001)). Among all possible objectives, the most widely used in scheduling parallel machines is the makespan minimization. Indeed, the makespan minimization allows the maximization of machine utilization and provides a good load-balance. The makespan minimization is denoted as $\gamma = C_{max}$.

In general, the classical scheduling problems in parallel machines assume that:

- Each job cannot be processed in more than one machine simultaneously;
- A job once started should be completed;
- Processing time does not depend on the sequence;
- The job can wait for a free machine;
- No machine can process more than a job at a time;
- The number of jobs n is known and fixed;
- The number of machines m is known and fixed;
- Processing times are known and certain;
- During the schedule, no machine breakdowns are considered.

When the processing requirement of a job is considered as its total demand (*e.g.* lot), we can split the jobs into sub-jobs (*e.g.* sub-lots) that can be processed independently on the machines. Under splitting, we accept to overlap the sub-jobs so as to finish the processing requirement as soon as possible. Potts and Van Wassenhove (1992) referred to this process as lot-streaming or lot-sizing and Xing and Zhang (2000)

denote the job characteristic as $\beta = Split$. When the processing of a job can be interrupted several times and later resumed but, the sub-jobs of the same job cannot be processed in parallel, we denote $\beta = pmtn$ to mean preemption.

In the classical scheduling algorithms, it is assumed that the scheduler has full information about the problem instance before the process of scheduling actually starts (*e.g.* the number of machines, the number of jobs, the processing times are known with exactitude). However, the uncertainty of processing times is prevalent in the applications that involves such problems. In Section 3.1, we describe the problems $Rm|Split|C_{max}$ and $Rm|pmtn|C_{max}$ and we show the limits of the classical deterministic algorithms in solving these problems under uncertain processing times. Besides, we propose a preliminary approach that aims to enforce the feasibility of the schedule under different scenarios by tolerating the violations of some processing requirements, and we show that under this approach the overproduction or the shortage can be very important. To avoid this, we propose in Section 3.2 an artificial scenario based approach under which we construct a set of feasible solutions based on a set of generated artificial scenarios. We choose the robust solutions based on an evaluation algorithm that computes a robustness measure of each solution. In Section 3.3, we report the results of the computational experiments respectively in splitting and preemptive cases.

3.1 The need for robustness under uncertain processing times

3.1.1 Makespan minimization on unrelated parallel machines under splitting and preemption

The makespan minimization on unrelated parallel machines with splitting denoted $Rm|Split|C_{max}$ is defined as follows. n independent jobs are allowed to be processed by any of the m unrelated parallel machines. The processing times p_{ij} depends on job j and machine i. Each job can be split into continuous sub-jobs and processed independently on the m machines with allowed parallelism so as to finish the processing of all demands as soon as possible. For a high utilization of the machines and a good load-balancing, the objective is to determine the optimal schedule so as to minimize the makespan C_{max}. According to Xing and Zhang (2000), this process is referred to as lot-sizing or lot-streaming and the split parts as continuous sub-lots. The deterministic problem $Rm|Split|C_{max}$ was formulated by Xing and Zhang (2000) as a linear problem (LP_{Split}):

$$\text{Minimize } C_{max} \tag{3.1}$$

Subject to

$$\sum_{i=1}^{m} \frac{t_{ij}}{p_{ij}} = 1, \qquad j = 1, \ldots, n \tag{3.2}$$

$$\sum_{j=1}^{n} t_{ij} - C_{max} \leq 0, \qquad i = 1, \ldots, m \tag{3.3}$$

$$0 \leq t_{ij}, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n \tag{3.4}$$

where each temporal variable t_{ij} is the total amount of time spent by job j on machine i and C_{max} is the makespan which is the maximum job completion time over the m machines. The objective function (3.1) in (LP_{Split}) minimizes the makespan. Constraints (3.2) ensure that each job j receives the required amount of processing, while constraints (3.3) enforce that the total processing on each machine i is less or equal to the makespan. Lastly, constraints (3.4) imply that each temporal variable t_{ij} is positive.

The given formulation can be solved in polynomial time. An optimal solution which is a matrix of durations T allows to construct an optimal schedule. Indeed, all the sequences are feasible.

In the preemptive case, the makespan minimization on unrelated parallel machines, denoted $Rm|pmtn|C_{max}$, is as an extension of the split case. The major difference between the preemptive problem and the split one is that in the makespan minimization on unrelated parallel machines with preemption, the computation of a sequence is also a part of the decision. In fact, each job can be split into continuous sub-jobs that can be processed independently on the m machines without parallelism (no overlapping). Lawler and Labetoulle (1978) have suggested a two step algorithm to solve $Rm|pmtn|C_{max}$. The first step consists in solving a linear formulation (LP_{pmtn}) given as follows:

$$\text{Minimize } C_{max} \tag{3.5}$$

Subject to

$$\sum_{i=1}^{m} \frac{t_{ij}}{p_{ij}} = 1, \qquad j = 1, \ldots, n \tag{3.6}$$

$$\sum_{j=1}^{n} t_{ij} - C_{max} \leq 0, \qquad i = 1, \ldots, m \tag{3.7}$$

$$\sum_{i=1}^{m} t_{ij} - C_{max} \leq 0, \qquad j = 1, \ldots, n \tag{3.8}$$

$$0 \leq t_{ij}, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n \tag{3.9}$$

where each temporal variable t_{ij} is the total amount of time spent by job j on machine i and C_{max} is the makespan. Constraints (3.6) ensure that each job j is receiving the required amount of processing. The constraints (3.7) ensure that the (workload) total amount of processing on each machine i is less or equal to C_{max} while the constraints (3.8) also insures that the total processing time of each job do not exceed the makespan. The constraints (3.8) prohibit the overlap between job parts of the same job. Lastly, constraints (3.9) imply that variable t_{ij} is positive. The objective (3.5) minimizes the makespan.

The linear formulation can be solved in polynomial time, but the solution of (LP_{pmtn}) does not prescribe an actual schedule. The second step, then, consists of the construction of the optimal schedule with the information provided by (LP_{pmtn}) resolution. Indeed, for any feasible solution to the linear program, there exists a feasible schedule with the same values of t_{ij} and C_{max}. To prove this assertion, Lawler and Labetoulle (1978) solved the preemptive open-shop scheduling problem $Om|pmtn|C_{max}$ defined by Gonzalez and Sahni (1976). It comes back to consider that once the durations and the corresponding makespan are

computed, the problem becomes a sequencing problem of a set of jobs that are composed of a number of tasks (that we call the sub-jobs) where no restrictions are placed in the order in which the tasks (sub-jobs) of any job should be processed. The only constraint to respect is that a machine can process only one job at a time and a job can be worked by only one processor at a time. There is no restriction on the order in which a given job can be worked on by the different processors, or on the order in which a given processor can work on jobs (Hence the term "open shop."). However, the construction of a sequence might require additional preemption. It means that the t_{ij}s once assigned to a given machine can be interrupted and resumed later in that machine. According to Gonzalez and Sahni (1976), $Om|pmtn|C_{max}$ is solvable in polynomial time. An analysis of computations done by Gonzalez (1979) has shown that the $Om|pmtn|C_{max}$ is solvable in $O(r + \min(r^2, n^4, m^4))$ where r is the number of nonzero elements in T the matrix of durations t_{ij}'s. The maximum number of preemptions introduced is $min\{rn, rm, n^3, m^3\}$. And according to Lawler and Labetoulle (1978), an upper bound on the number of preemptions required for a C_{max}-$optimal$ schedule on unrelated parallel machines is $4m^2 - 5m + 2$.

3.1.2 Limitations of the classical algorithms under uncertain processing times

In practice, the processing times p_{ij}s are not often certain. In fact, each processing time p_{ij} depends both on the processing requirement p_j of the job j representing its demand, and the speed v_{ij} of machine i to process the job j. Each p_{ij} can be decomposed as: $p_{ij} = p_j/v_{ij}$. If the demands are uncertain, which is often the case in reality, the processing times become also uncertain. Under these uncertainties, the solving approaches introduced in the previous sections, LP_{Split} and the 2 step algorithm of Lawler based on LP_{pmtn}, may lead to schedules that cannot be executed as planned: when applied to the actual realization of processing times, the optimal schedules based on the nominal instance of processing times become sub-optimal and sometimes infeasible. To point out this claim we will consider the following example:

Example 3.1.1 *We consider an example in which 4 jobs are to be processed by 3 unrelated parallel machines. The v_{ij}s are certain and depend on both machines and jobs as given in Table 3.1.*

We consider a forecast instance of p^n_{ij}s called the nominal instance I^n (Table 3.2) corresponding to the demand: $D^n = (30, 50, 30, 100)$.

TABLE 3.1: Matrix of speeds

v_{ij}	J_1	J_2	J_3	J_4
M_1	3	2	4	1
M_2	3	1	3	2
M_3	3	4	4	4

a- In the splitting case: *according to LP_{Split} resolution, Job 1 and Job 3 are entirely processed on machine 1 for 10 units and 7.5 units respectively, Job 2 is processed for 5.62 units on machine 1 and for 9.69 units on machine 3, and Job 4 is processed for 23.12 units on machine 2 and for 13.44 units on machine 3. The optimal makespan, which is equal to 23.12 units, features a high utilization of the three machines with*

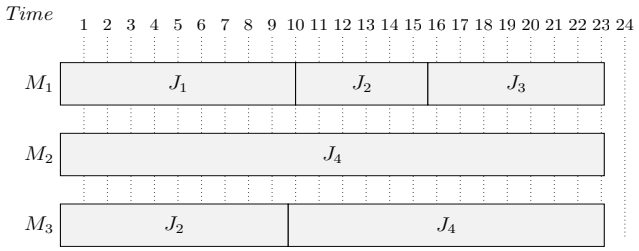
TABLE 3.2: Nominal Instance

p_{ij}^n	J_1	J_2	J_3	J_4
M_1	30/3	50/2	30/4	100/1
M_2	30/3	50/1	30/3	100/2
M_3	30/3	50/4	30/4	100/4

TABLE 3.3: Real instance

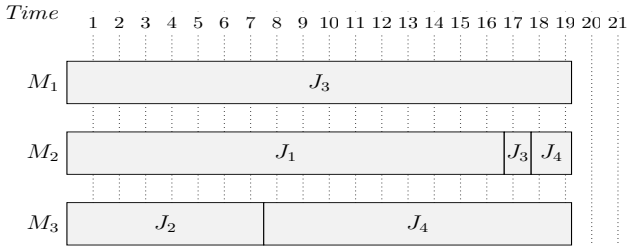
p_{ij}^r	J_1	J_2	J_3	J_4
M_1	50/3	30/2	80/4	50/1
M_2	50/3	30/1	80/3	50/2
M_3	50/3	30/4	80/4	50/4

FIGURE 3.1: An optimal schedule of the nominal instance in the splitting case



a strict load balancing (Figure 3.1). But, under demand uncertainty, the realization of processing times will be different from the nominal instance. For a real demand given as: $D^r = (50, 30, 80, 50)$*, the real instance* I^r *of* p_{ij}^r*s is represented in Table 3.3. In that case, the decision-maker may have two basic strategies:*

FIGURE 3.2: An optimal schedule of the real instance in the splitting case



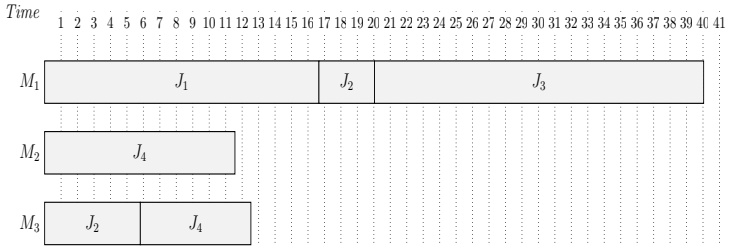
The first strategy is to adapt the schedule to the new instance I^r *by the computation of its corresponding optimal schedule. Accordingly, the four jobs are to be scheduled as shown in Figure 3.2: Job 1 is entirely processed on machine 2 for 16.67 units, Job 2 is to be entirely processed on machine 3 for 7.5 units, Job 3 is processed on machine 1 for 19.23 units and on machine 2 for 1.02 units, Job 4 is processed for only 1.53 units on machine 2 and for 11.73 units on machine 3. Nevertheless, this strategy is not stable because it requires new machine setup and changeover (even if we do not consider explicitly these costs), besides job handling to make the needed adjustments. The more different the structures of the two schedules are, the more important the adjustment efforts will be. These syndromes of schedule nervousness, as called by practitioners, can turn out to be a serious problem under big instances because they generate a considerable amount of efforts and hidden costs as well as a general loss of confidence in the scheduling function.*

The second strategy is to schedule the real instance according to the optimal nominal schedule. Therefore, we propose to calculate the machine participation ratios to process jobs according to the optimal schedule of the nominal instance as given in table of Figure 3.3: $\frac{t_{ij}^{n}*}{p_{ij}^{n}}$ and then we apply these ratios to schedule the real instance.

FIGURE 3.3: Schedule of the real instance according to the nominal instance solution in the splitting case

$\frac{t_{ij}^{n}}{p_{ij}^{n}}$	J_1	J_2	J_3	J_4
M_1	1	0.22	1	0
M_2	0	0	0	0.46
M_3	0	0.78	0	0.54

Participation ratios

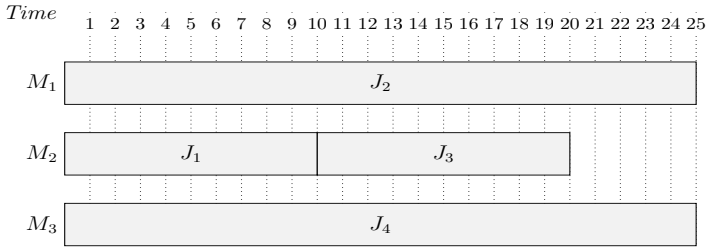


Schedule of the real instance

By applying the optimal nominal solution to schedule the real instance I^r we obtain a schedule that requires a makespan equal to 40.04 time units (Figure 3.3). The generated decision is suboptimal: the makespan C_{max} has increased over 108% time units compared to the optimal makespan of the real scenario C_{max}^{r*}. Besides the load balancing is altered as the machine M_1 is overused while machines M_2 and M_3 are more than half schedule time in the idle time.

b- In the preemptive case: *The observations are similar to those in the splitting case, i.e. when we ignore the uncertainty, the optimal preemptive schedule based on the nominal instance become vulnerable when applied to the real instance.*

FIGURE 3.4: An optimal schedule of the nominal instance in the preemptive case



According to the resolution of (LP_{pmtn}), the preemptive optimal schedule of the nominal instance I^n can handle all jobs with a makespan equal to 25 time units (see Figure 3.4). As we can observe, for this particular instance, each job was completely assigned to only one machine without any preemption: Job 2 is processed on machine 1 and Job 4 is processed on machine 3 for 25 units respectively, whereas Job 1 and job 3 are processed on machine 2 for 10 units respectively. Consequently, there is no need to solve the corresponding preemptive open shop in order to construct the schedule.

FIGURE 3.5: An optimal schedule of the real instance in the preemptive case

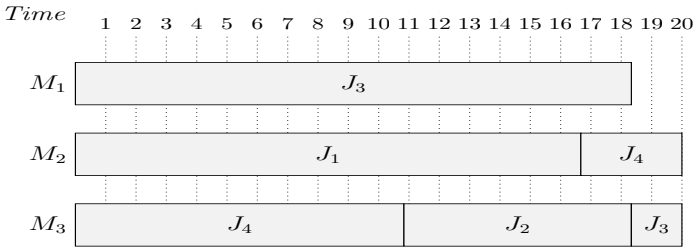
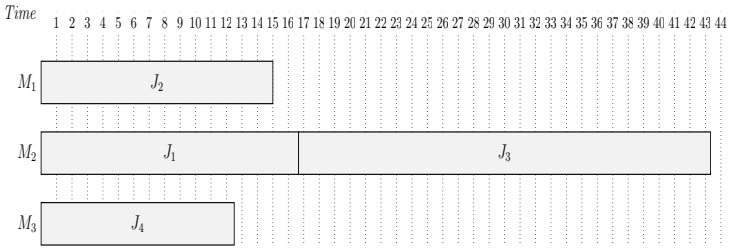


FIGURE 3.6: Schedule of the real instance according to the nominal instance solution in the preemptive case

$\frac{t_{ij}^{n*}}{p_{ij}^{n}}$	J_1	J_2	J_3	J_4	$\prec$
M_1	0	1	0	0	$J2$
M_2	1	0	1	0	$J1 \prec J3$
M_3	0	0	0	1	$J4$

Participation ratios



Schedule of the real instance

If we again consider the real instance of processing times given in (Table 3.3), the two basic strategies have weakness points. When we totally adapt the schedule to the real instance, we construct an optimal schedule which is very different from the nominal schedule (see Figure 3.5) under which Job 1 is processed for 16.67 units on machine 2, Job 2 is processed for 7.5 units on machine 3, Job 3 is processed for 18.32 units on machine 1 and for 1.66 units on machine 3, and Job 4 is processed for 3.25 units on machine 2 and for 10.75 units on machine 3. The makespan is equal to 20 *units. But, as we can observe, the total adaptation can imply high deviations in terms of assignments, starting times and completion times. Consequently, the nervousness aspects can be accurate under a totally adaptive strategy in the preemptive case.*

When we schedule the real instance based on the permutation and the machine participation ratios to process jobs according to the optimal schedule of the nominal instance (see Figure 3.6), we notice that the degradation of the makespan is very significant (see Figure 3.6).

The example shows that a nominal schedule which is determined to be optimal regarding the makespan criterion prior to its execution can turn to be vulnerable under minor or serious uncertainty. Furthermore, the calculation of a new solution under each new scenario can also lead to hidden costs due to nervousness aspects.

The major weakness of the classical scheduling approach, when the processing times are uncertain, is that the schedules which are optimal with respect to the nominal instance (forecast instance) might be substantially infeasible or yield poor performance when evaluated relatively to the actual processing times. For

decision makers that have to face the consequences regardless of the realized instance, schedule infeasibility or sub-optimality are unacceptable, even though that schedule is the "optimal" one for the nominal scenario.

It is generally admitted that forecast values are usually wrong. Any scheduling algorithm that ignores this fact, and is focused on the "nominal" instance of forecast processing times, is unacceptable for decision environments subject to uncertainty. As shown in the example, the schedules which are determined to be optimal based on the nominal instance can lead to poor performances when applied to the real instance. Furthermore, the calculation of a new schedule under each new instance does not provide a baseline schedule and requires adjustment efforts. In light of this, the scheduling approach to develop should be designed to take into account the uncertainty of processing times.

Stochastic scheduling models adopt a probabilistic viewpoint, treating uncertain processing times as random variables. Under this scheme, a probability distribution of the processing times should be available and the scheduling model is formulated with the objective to optimize the expected makespan. However, in many cases, such a distribution is not available.

When schedulers are confronted with a significant uncertainty of processing times that cannot be approximated with a probability distribution, many authors claimed that non-probabilistic discrete scenarios offer a good representation of uncertainty (*e.g.* Kouvelis and Yu (1997), Averbakh (2006), Aissi *et al.* (2011), Kasperski *et al.* (2012), Roy (2010), *etc.*). As discussed earlier in Chapter 1 and Chapter 2, the discrete scenarios are suitable in situation of ambiguity when there is a lack of information about the probabilities and when the schedulers aim to be protected against any potential future. They are widely used to represent the uncertain processing times in scheduling problems especially when the uncertainty is due to uncertain demand. Furthermore, many approaches called robust result from the use of discrete scenarios. Under these approaches, we generally seek for solutions that optimize a global performance instead of seeking for solutions that optimize a local performance as the performance of the decision is evaluated across all the potential scenarios.

From the foregoing, we assume that the uncertainty of processing times can be represented by a set of discrete scenarios $\Omega_k = \{s_1, s_2, \ldots, s_k\}$. Each scenario $s \in \Omega_k$ represents a potential realization of the uncertain processing times and captures a situation that might occur. We denote by p_{ij}^s the processing time of job j on machine i under scenario s, and C_{max}^{s*} the optimal makespan under scenario s. The nominal scenario is also included in the set Ω_k which makes the deterministic formulation only a special case.

Based on the deterministic linear formulation of $Rm|Split|C_{max}$ (*resp.*. $Rm|pmtn|C_{max}$), the uncertainty of processing times is affecting the data in the equality constraints. Thus, the robust problem is equivalent to a constraint feasibility problem where the objective is to find the schedule that is feasible under any scenario as expressed in the constraints Equation 3.10.

$$\sum_{i=1}^{m} \frac{t_{ij}}{p_{ij}^s} \quad = 1, \quad j = 1, \ldots, n, \; s = s_1, \ldots, s_k. \tag{3.10}$$

Under this formulation, the set of solutions may be empty since the equality constraints cannot be insured for different scenarios.

To overcome the infeasibility, a first solution consists in adding slack variables in the equality constraints as suggested in Mulvey *et al.* (1995). This approach that we call "slack based approach" is explained in details in the next section.

3.2 Slack based approach under discrete processing time scenarios

Mulvey *et al.* (1995) suggested a general robust optimization framework for mathematical programs with continuous variables. This framework integrates a goal programming formulation with a scenario-based representation of uncertain problem data. The model based on Mulvey's framework recognizes that it may not always be possible to get a feasible solution to a problem under all the potential scenarios. Therefore, the model through the use of error terms and penalty function allows to find a solution that violates the constraints by the least amount. Its objective is to compute a solution that remains "almost" feasible for any realization of s: a model robust solution. When we assume a discrete scenario representation of the processing times, based on Mulvey's approach, we propose a robust reformulation that admits the violation of these constraints. For this purpose, we add slack variables in the equality constraints as follows:

$$\sum_{i=1}^{m} \frac{t_{ij} + z_{ij}^{s+} - z_{ij}^{s-}}{p_{ij}^{s}} = 1, \qquad j = 1, \ldots, n,\ s = s_1, \ldots, s_k \tag{3.11}$$

We distinguish two kinds of error terms in (3.11):

- The z_{ij}^{s+} express the necessity of a replenishment to reach the processing times required by jobs under each scenario (case of under-production).

- The z_{ij}^{s-} compensate an overrun of the processing times under each scenario (case of overproduction).

We control the violations due to the slacks through a weighted penalty function $\rho(.)$:

$$\rho(.) = \frac{w}{k} \sum_{s=s_1}^{s_k} Z_{max}^{s} \qquad or \qquad \rho(.) = w. \max_{s \in \{s_1, \ldots, s_k\}} Z_{max}^{s} \tag{3.12}$$

where Z_{max}^{s} is the maximal violation induced by the use of slacks under a scenario s, w is the weight of the penalty and k the number of potential scenarios.

One can choose over different strategies:

- Case 1: Z_{max}^{s} is the maximal violation over machines computed according to the following constraints:

$$\sum_{j=1}^{n} (z_{ij}^{s+} + z_{ij}^{s-}) \leq Z_{max}^{s}, \qquad i = 1, \ldots, m,\ s = s_1, \ldots, s_k, \tag{3.13}$$

- Case 2: Z_{max}^{s} is the maximal violation over jobs computed according to the following constraints:

$$\sum_{i=1}^{m} (z_{ij}^{s+} + z_{ij}^{s-}) \leq Z_{max}^{s}, \qquad j = 1, \ldots, n,\ s = s_1, \ldots, s_k, \tag{3.14}$$

- Case 3: Z^s_{max} is the maximal violation over jobs and machines computed according to the following constraints

$$z^{s+}_{ij} + z^{s-}_{ij} \leq Z^s_{max}, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n,\ s = s_1, \ldots, s_k \tag{3.15}$$

- Case 4: Z^s_{max} is the total of all violations over jobs and machines computed as

$$\sum_{i=1}^{m}\sum_{j=1}^{n}(z^{s+}_{ij} + z^{s-}_{ij}) = Z^s_{max}, \qquad s = s_1, \ldots, s_k \tag{3.16}$$

The objective is to: Minimize C_{max} + $\rho(.)$ The minimization of the makespan C_{max} aims to optimize the performance of the schedule while the penalty function $\rho(.)$ aims to minimize the violation over all the potential scenarios (average or max) .

Here, we present some cases of the slack based approach to compute model robust schedules in the case of the makespan minimization on unrelated parallel machines with splitting (*resp.*. preemption). We can propose different formulations.

Formulation 1:

$$\text{Minimize } C_{max} + \tfrac{w}{|k|}\textstyle\sum_{s=1}^{k} Z^s_{max} \tag{3.17}$$

Subject to

$$\sum_{i=1}^{m}\frac{t_{ij} + z^{s+}_{ij} - z^{s-}_{ij}}{p_{ij}} = 1, \qquad j = 1, \ldots, n,\ s = s_1, \ldots, s_k \tag{3.18}$$

$$\sum_{j=1}^{n} t_{ij} - C_{max} \leq 0, \qquad i = 1, \ldots, m \tag{3.19}$$

$$\sum_{i=1}^{m}(z^{s+}_{ij} + z^{s-}_{ij}) \leq Z^s_{max}, \qquad j = 1, \ldots, n,\ s = s_1, \ldots, s_k \tag{3.20}$$

$$0 \leq t_{ij} + z^{s+}_{ij} - z^{s-}_{ij}, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n,\ s = s_1, \ldots, s_k \tag{3.21}$$

$$0 \leq t_{ij}, z^{s+}_{ij}, z^{s-}_{ij}, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n,\ s = s_1, \ldots, s_k \tag{3.22}$$

Formulation 2:

$$\text{Minimize } C_{max} + \tfrac{w}{|k|}\textstyle\sum_{s=1}^{k} Z^s_{max} \tag{3.23}$$

Subject to 3.18, 3.19, 3.21, 3.22 and

$$\sum_{i=1}^{m}\sum_{j=1}^{n}(z^{s+}_{ij} + z^{s-}_{ij}) \leq Z^s_{max}, \qquad s = s_1, \ldots, s_k \tag{3.24}$$

In the formulations 1 and 2, we minimize the sum of the makespan C_{max} and the total weighted violation over

all the scenarios $\sum_{s=1}^{k} Z_{max}^{s}$. The slack variables in the constraints (3.18) allow the feasibility of the equality under all the scenarios. Constraints (3.21) combined with the objective function imply the computation of non-null matrix T. In the formulation 1, Z_{max}^{s} is computed as the total violation over jobs under the scenario s (*c.f.* constraints (3.20)) while in the formulation 2 it is computed as the total violation by machines and jobs (*c.f.* constraints (3.24)). The temporal variables t_{ij} and the slack variables z_{ij}^{s+}, z_{ij}^{s-} are positive (3.22).

Formulation 3:

$$\text{Minimize } C_{max} + Z_{max} \tag{3.25}$$

Subject to 3.18, 3.19, 3.21, 3.22 and

$$\sum_{i=1}^{m}\sum_{j=1}^{n}(z_{ij}^{s+} + z_{ij}^{s-}) \leq Z_{max}^{s}, \qquad s = s_1, \ldots, s_k \tag{3.26}$$

$$Z_{max}^{s} \leq Z_{max}, \qquad s = s_1, \ldots, s_k \tag{3.27}$$

In the formulation 3, we minimize the sum of the makespan C_{max} and the maximal violation over all the scenarios Z_{max}. $Z_{max} = \max_{s=1}^{k} Z_{max}^{s}$ is computed as the maximal total violation over jobs and machines, over the scenario s (*c.f.* constraints (3.26),(3.27)).

In order to extend the formulations to the preemptive case, one should add the constraints of non overlapping (Equation 3.8) .

We illustrate the proposed approach through the example initiated in section 3.1.1.

Example 3.2.1 *We assume that uncertain processing times are due to uncertain demands. We represent the uncertainty of demand by 5 potential scenarios corresponding to the future realizations (Table 6.1). The matrix of job-machine dependant speeds v_{ij} is invariant (Table 3.5).*

TABLE 3.4: Potential scenarios of demand

	p_1	p_2	p_3	p_4
s_1	30	50	30	100
s_2	50	30	80	50
s_3	40	50	100	80
s_4	35	40	50	70
s_5	40	60	100	50
s_6	60	50	80	70

TABLE 3.5: Matrix of speeds

	J_1	J_2	J_3	J_4
M_1	3	2	4	1
M_2	3	1	3	2
M_3	3	4	4	4

a- In the case of splitting: *The resolution of the slack based approach applied to $Rm|Split|C_{max}$ gives the following results. Under formulation 1, for $w = 6$, the produced quantities according to the robust schedule are: $(48.75, 35, 75, 55)$ with a makespan equal to 20 units (see Table 3.6). The demand deviations (%) due to the error terms are represented in Table 3.7. This shows that the over productions and the under productions are important for all the scenarios except for s_2.*

TABLE 3.6: A slack based schedule based on formulation 1

t_{ij}	J_1	J_2	J_3	J_4
M_1	1.25	0	18.75	0
M_2	15	0	0	5
M_3	0	8.75	0	11.25

TABLE 3.7: Job processing requirement deviations in the splitting case based according to formulation 1 resolution

	p_1	p_2	p_3	p_4
s_1	+62.5%	-30%	+150%	-45%
s_2	-2.5%	+16.6%	-6.25%	-10%
s_3	+21.87%	-30%	-25%	-43.75%
s_4	+39.2%	-12.5%	+50%	-21.42%
s_5	+21.87%	-41.66%	-50%	+10%
s_6	-18.75%	-30%	-6.25%	-21.42%

Under formulation 2, for $w = 6$, the produced quantities are: $(40, 50, 80, 70)$ with a makespan equal to 23.33 time units (Table 3.8). The produced demand deviations are given in Table 3.9.

TABLE 3.8: A slack based schedule according formulation 2 resolution

	J_1	J_2	J_3	J_4
M_1	3.33	0	20	0
M_2	10	0	0	13.33
M_3	0	12.5	0	10.83

TABLE 3.9: Job processing requirement deviations in the splitting case based on formulation 2

	p_1	p_2	p_3	p_4
s_1	+33.33%	0%	+16.66%	-30%
s_2	-20%	+66.6%	0%	+40%
s_3	0%	0%	-20%	-12.5%
s_4	+14.28%	+25%	-60%	0%
s_5	0%	-16.66%	-20%	+40%
s_6	-33.33%	0%	0%	0%

TABLE 3.10: Model robust solution in the preemptive case based on formulation 2

	J_1	J_2	J_3	J_4
M_1	1.66	1.66	20	0
M_2	11.66	0	0	11.66
M_3	0	11.66	0	11.66

b- In the case of preemption: *For $w = 6$, the resolution of formulation 1 leads to the same schedule and the same quantities as in the case of splitting. In contrary, the resolution of formulation 2 leads to an equivalent solution with a makespan that is also equal to 23.33 units (Table 3.10). This solution is also*

a solution to the case of splitting. The jobs are assigned to the machines with different durations, but the produced theoretical quantities are the same as in the case of splitting: $(40, 50, 80, 70)$. *Consequently, the over productions and the under productions are the same as those obtained in Table 3.9.*

We can notice from the example that the overproduction and the underproduction quantities are important for all the scenarios. Nevertheless, the penalty function in formulation 2 lead to better results than the penalty function in formulation 1. The slack based formulations lead to solutions that correspond to scenarios which are not in the set of the potential realizations.

When the demand is uncertain and its satisfaction is a must, accepting overproduction or underproduction does not solve the real problem but only a relaxed version of this problem. Therefore, this modelling approach cannot be used in all activity sectors because overproduction and underproduction are not often accepted in practice. Ben-Tal *et al.* (2009) claimed that one should avoid whenever possible using slack variables in order to ensure the feasibility of the constraints. Therefore, we will present in section 3 another approach that provides feasible solutions under the discrete processing time scenarios without introducing slack variables. This approach called artificial scenario solution based approach is a constructive approach designed for decision-makers interested in using simple heuristics to generate robust solutions.

3.3 Artificial scenario solution based approach under discrete processing time scenarios

In contrary to the slack based approach, the artificial scenario solution based approach aims to construct a set of feasible solutions without introducing slack variables.

The artificial scenario solution approach is based on scenario analysis, is a form of projection under which we evaluate a solution under a scope of possible futures instead of considering a single exact future. The uncertainty of processing times is represented by discrete scenarios that reflect the future data realizations. And, the scope of their related future decisions is observed and evaluated.

3.3.1 The global approach description

The artificial scenario solution based approach is divided into four successive steps as described in algorithm 1.

Algorithm 1: Artificial scenario solution approach

Data:

Ω_k: the set of potential scenario.

1 Construct the set of artificial scenarios of processing times Ω_k^a from Ω_k the set of potential scenarios;
2 Construct the set of feasible solutions based on the artificial scenario solutions;
3 Compute the local performances of the artificial scenario solutions;
4 Compute the robustness measures of the artificial scenario solutions and their ranking;

In *Step 1*, from the set of potential scenarios Ω_k, we construct a set of artificial scenarios Ω_k^a. We consider scenarios that are commonly used in practice by decision-makers. These artificial scenarios are:

- s_{max} the scenario in which the processing times take the maximal values among all the values of the potential scenario processing times in Ω_k,
- $s_{average}$ the scenario in which the processing times take the average values among all the values of the potential scenario processing times in Ω_k, it represents the called "forecast" scenario,
- s_{median} the median scenario in which the processing times are computed by sorting the values of the potential scenario processing times in Ω_k, and picking the middle ones,
- s_{min} the min scenario in which processing times take the minimal values among all the values of the potential scenario processing times in Ω_k,

The term artificial is used to mean that these scenarios do not necessarily belong to the set of potential scenarios.

The artificial worst-case scenario s_{max} is the scenario in which processing times take the maximal values among all the values of the potential scenarios. This scenario can inherently provide temporal protection due to the fact that the processing times are extended, and it can provide flexibility thanks to the offered margins. But, the worst-case scenario is often viewed as pessimistic.

The average scenario $s_{average}$ can be used as an alternative scenario to s_{max} since it increases only the processing times under the average while it reduces the processing times that are over the average. However, the average values do not reflect the tendency of the distribution. For this reason, we proposed also to compute the median scenario s_{median}. The processing times of s_{median} are values that divide the list of processing times among Ω_k into a list of higher half and a list of lower half. $p_{ij}^{s_{median}}$ is found by arranging all the values from lowest value to highest value and picking the middle one.

s_{min} is the scenario in which the processing times take the minimal processing times values over all the potential scenarios values. It represents the best-case scenario in opposite to s_{max}. And finally, we construct random scenarios by picking processing time values on the potential scenario processing times, and we denote:

- s_{random} the scenario of randomly picked processing times in Ω_k,

The objective is to compare the artificial scenario solutions that are commonly used in practice between them, and also to compare them to random solutions. For s_{max} solution and s_{median} solution particularly, some optimality (or approximation) results are provided in robust scheduling literature (seeAloulou and Della Croce (2008), Kasperski and Zieliński (2016)). We will verify if these properties can be extended to the studied problems.

In ***Steps 2 and 3***, we generate a set of feasible solutions based on the artificial scenario solutions, and we compute their local performances when applied to each potential scenarios $s \in \Omega_k$.

An artificial scenario solution is composed of two decisions which are the matrix of assignments and the job permutations on machines. We will distinguish the case of splitting and the case of preemption:

a- In the case of splitting, for each artificial scenario s_a, we solve the deterministic formulation LP_{Split} by considering the processing time of the artificial scenario as data entry of the linear program. This leads to a matrix durations $T^{s_a^*}$. Then, we compute the assignment ratio of each job j to each machine i according to the considered artificial scenario solution such as:

$$x_{ij}^{s_a^*} = \frac{t_{ij}^{s_a^*}}{p_{ij}^{s_a}}, \qquad \forall s_a \in \Omega_k^a \tag{3.28}$$

We choose any feasible permutation to construct the optimal schedule. The algorithm 2 gives the example of s_{max} solution computation under splitting.

Algorithm 2: Computation of the artificial scenario s_{max} solution in the case of splitting

Data:
$p_{ij}^{s_{max}}$: processing times under the artificial scenario s_{max}
Result: $X^{s_{max}^*}$ optimal assignment ratio matrix of s_{max}
1 Solve (LP_{Split}) by considering the processing times ($p_{ij}^{s_{max}}$) of the scenario s_{max} ;
$T^{s_{max}} \leftarrow$ optimal duration matrix of scenario s_{max};
2 Compute the ratio of participation of each machine i to the realization of a job j under the considered artificial scenario solution
foreach $(i \in M, j \in N)$ **do**
  $x_{ij}^{s_{max}^*} \leftarrow \frac{t_{ij}^{s_{max}}}{p_{ij}^{s_{max}}}$
end
return $X^{s_{max}^*}$

Let M be the set of m machines and N the set of n jobs. For each artificial scenario solution, we calculate the local performances of the potential scenarios under this solution. For this purpose, we schedule each potential scenario $s \in \Omega_k$ according to the assignment ratios of $X^{s_a^*}$ and then we compute its makespan under $X^{s_a^*}$

$$C_{max}^s(X^{s_a^*}) = \max_{i \in M} \sum_{j=1}^{n} p_{ij}^s x_{ij}^{s_a^*}, \tag{3.29}$$

and its regret

$$C_{max}^s(X^{s_a^*}) - C_{max}^{s*} \tag{3.30}$$

where C_{max}^{s*} is the optimal makespan of the potential scenario s.

b- In the case of preemption, for each artificial scenario s_a, we solve the deterministic formulation LP_{pmtn} by considering the processing times of the artificial scenario as data entry of the linear program. We compute the ratio of participation of each machine i to the realization of a job j (assignment ratios) under the considered artificial scenario solution as given in (3.28).

The algorithm 3 describes the computation of the artificial scenario s_{max} solution under preemption:

Algorithm 3: Computation of the artificial scenario s_{max} based solution in the preemptive case

Data:
$p_{ij}^{s_{max}}$: processing times under the artificial scenario s_{max}
Result: $X^{s^*_{max}}$ optimal assignment ratio matrix of s_{max}
1 Solve (LP_{pmtn}) by considering the processing times $(p_{ij}^{s_{max}})$ of the scenario s_{max} ;
$T^{s_{max}} \leftarrow$ optimal duration matrix of scenario s_{max};
2 Compute the ratio of participation of each machine i to the realization of a job j under the considered artificial scenario solution
foreach $(i \in M, j \in N)$ **do**
$x_{ij}^{s_{max}} \leftarrow \frac{t_{ij}^{s_{max}}}{p_{ij}^{s_{max}}}$
end
return $X^{s^*_{max}}$

As the sequence is part of the decision in the preemptive case, we solve the open shop corresponding to $T^{s^*_a}$ according to (Gonzalez and Sahni (1976)). We then deduce a permutation $\sigma^{s^*_a}$ and we denote:

$$\sigma_i^{s^*_a}(j) = j', \; if \; j \; is \; scheduled \; before \; j' \; on \; machine \; i. \tag{3.31}$$

We could schedule each potential scenario $s \in \Omega_k$ according to the optimal schedule of s_a, *i.e.* we assign the jobs according to the assignment ratio solution $X^{s^*_a}$ and we sequence the jobs with respect to the permutation $\sigma^{s^*_a}$. To correct the jobs overlapping, we use the right-shift algorithm introduced in Abumaizar and Svestka (1997). The second alternative is to compute the durations of the potential scenarios according to the artificial scenario assignment ratios and then construct for each scenario a feasible sequence by solving a preemptive open shop. The sequence is not necessary the same for all scenarios.

A feasible schedule for each scenario is then obtained: - $p_{ij}^s x_{ij}^{s^*_a}$ is the sub-job j duration in machine i under scenario s according to s_a assignment,

- $s_{ij}^s(X^{s^*_a})$ is the starting time of job j in machine i under scenario s.

We compute the maximal completion time of scenario s such as:

$$C_{max}^s(X^{s^*_a}) = \max_{i \in M} \max_{j \in N}(s_{ij}^s(X^{s^*_a}) + p_{ij}^s x_{ij}^{s^*_a}). \tag{3.32}$$

Finally, in **Step 4**, we evaluate the robustness of the computed solutions under the set of discrete scenarios Ω_k. We use the classical robustness measures which are the worst-case makespan and the maximal regret under both splitting and preemption.

The worst-case makespan of the artificial scenario solution $X^{s^*_a}$ across all the scenarios of Ω_k measures the maximal cost of $X^{s^*_a}$. It is defined as

$$\max_{s \in \Omega_k}\{C_{max}^s(X^{s^*_a})\}, \tag{3.33}$$

And the maximal regret of the artificial scenario solution $X^{s_a^*}$ across all the scenarios of Ω_k measures the maximal deviation from the optimal makespans of the potential scenarios. It is defined as:

$$\max_{s\in\Omega_k}\{C^s_{max}(X^{s_a^*}) - C^{s*}_{max}\}, \tag{3.34}$$

Let us consider the same example as given in the previous section to implement the artificial scenario solution based approach under both splitting and preemption:

Example 3.3.1 *In the example 3.2.1, we represented the uncertainty of demands by 6 different scenarios corresponding to the future realizations of demands (Table 6.1), the matrix of speeds v_{ij} is invariant (Table 3.5).*

TABLE 3.11: Optimal makespans of the potential scenarios in the splitting case

	s_1	s_2	s_3	s_4	s_5	s_6
C^{s*}_{max}	23.12	19.23	25.83	19.79	23.46	25.0

***Step 1**: From the set of potential scenarios Ω_6, we construct the set of artificial scenarios: we represent in Table 3.12 the job processing requirements (demands) of the artificial scenarios. The matrix of speeds is invariant.*

TABLE 3.12: Job processing requirements of the artificial scenarios

Ω^a_k	p_1	p_2	p_3	p_4
s_{max}	60	60	100	100
$s_{average}$	42.5	46.66	73.33	70
s_{median}	40	50	80	70
s_{min}	35	30	30	50
s_{random}	40	60	100	50

a- In the splitting case:

***Step 2**: For each artificial scenario, we solve using CPLEX the corresponding LP_{Split} model to compute the optimal durations and then we compute the assignment ratios. These decisions are reported in Table 3.13.*

***Step 3**: We schedule the potential scenarios according to the artificial scenario schedules (see C and Appendix C). We compute the local performances: the makespan and the regret of each potential scenario scheduled according to the artificial scenario schedule (Table 3.14).*

To compute the regrets, we compute the optimal makespans of the potential scenarios:

***Step 4:** We compute the global performances, i.e. the worst-case makespans and the maximal regrets over the potential scenarios (Table 3.15). In this example, the ranking of the artificial scenario solutions according to the global performances shows that if the decision-maker aims to minimize the worst-case*

TABLE 3.13: Optimal schedules of the artificial scenarios in the splitting case: durations (ratios)

X_{max}	J_1	J_2	J_3	J_4
M_1	6.25 (0.31)	0.0 (0.0)	25.0 (1.0)	0.0 (0.0)
M_2	13.75 (0.69)	0.0 (0.0)	0.0 (0.0)	17.5 (0.35)
M_3	0.0 (0.0)	15.0 (1.0)	0.0 (0.0)	16.25 (0.65)
$X_{average}$	J_1	J_2	J_3	J_4
M_1	4.37 (0.31)	0.0 (0.0)	18.33 (1.0)	0.0 (0.0)
M_2	9.79 (0.69)	0.0 (0.0)	0.0 (0.0)	12.91 (0.37)
M_3	0.0 (0.0)	11.66 (1.0)	0.0 (0.0)	11.04 (0.63)
X_{median}	J_1	J_2	J_3	J_4
M_1	3.33 (0.25)	0.0 (0.0)	20.0 (1.0)	0.0 (0.0)
M_2	10.0 (0.75)	0.0 (0.0)	0.0 (0.0)	13.33 (0.38)
M_3	0.0 (0.0)	12.5 (1.0)	0.0 (0.0)	10.83(0.62)
X_{min}	J_1	J_2	J_3	J_4
M_1	6.9 (0.69)	0.0 (0.0)	7.5 (1.0)	0.0 (0.0)
M_2	3.1 (0.31)	0.0 (0.0)	0.0 (0.0)	11.25 (0.45)
M_3	0.0 (0.0)	7.5 (1.0)	0.0 (0.0)	6.87 (0.55)
X_{random}	J_1	J_2	J_3	J_4
M_1	0.0 (0.0)	0.0 (0.0)	23.46 (0.93)	0.0 (0.0)
M_2	13.33 (1.0)	0.0 (0.0)	2.05 (0.07)	8.07 (0.33)
M_3	0.0 (0.0)	15.0 (1.0)	0.0 (0.0)	8.46 (0.67)

TABLE 3.14: Local performances of the artificial scenario solutions in the splitting case

	Measures	s_1	s_2	s_3	s_4	s_5	s_6
X_{max}	Makespan	28.75	25.20	29.16	21.37	29.16	26.25
	Regret	5.62	5.97	3.33	1.58	5.701	1.25
$X_{average}$	Makespan	28.27	25.14	29.11	21.04	29.11	26.74
	Regret	5.14	5.91	3.28	1.25	5.65	1.74
X_{median}	Makespan	27.97	24.16	28.33	22.08	28.33	28.33
	Regret	4.85	4.93	2.49	2.29	4.87	3.33
X_{min}	Makespan	26.25	31.456	34.16	20.52	34.16	33.75
	Regret	3.12	12.22	8.33	0.72	10.70	8.75
X_{random}	Makespan	29.42	26.38	28.30	24	23.46	32.95
	Regret	6.3	7.15	2.47	8.51	0	7.95

makespan or the regret, then the solution to choose is the artificial scenario s_{median} solution (X_{median}).

TABLE 3.15: Robustness measures of the artificial scenario solutions in the splitting case

Artificial scenario solution	Worst-case makespan	Maximal regret
X_{max}	29.16	5.97
$X_{average}$	29.11	5.91
X_{median}	28.33	4.93
X_{min}	34.16	12.22
X_{random}	32.95	8.51

But, we can notice that the gap between the global performances of s_{median}, $s_{average}$ and s_{max} is relatively low. We also compare the global performances of the robust artificial solutions and the optimal solution of the nominal scenario. The robustness measures of the nominal scenario solution given Table 3.16 show that the worst-case makespan and regret of $s_{nominal}$ are superior to those provided by the robust artificial scenario solutions.

TABLE 3.16: Local performances of the nominal scenario solution in the splitting case

X_{s1}	s_1	s_2	s_3	s_4	s_5	s_6
Makespan	23.12	40.04	43.95	28.66	45.08	45.62
Regret	0.0	20.81	18.12	8.875	21.625	20.62

See the schedules in Appendix C.

b- In the preemptive case:

Similarly, we apply the artificial scenario solution based approach to the preemptive problem $Rm|pmtn|C_{max}$ under uncertain processing times. Contrarily to the $Rm|Split|C_{max}$, the preemption requires to compute the sequence. We consider the same data example, as used in the previous section.

Example 3.3.2 *We have 6 scenarios of processing requirement and an invariant matrix of speed.*

TABLE 3.17: Optimal makespans of the potential scenarios in the preemptive case

	s_1	s_2	s_3	s_4	s_5	s_6
C^{s*}_{max}	25.0	20.0	25.88	20.0	25.0	25.0

Step 1*: The set of the artificial scenarios is the same as given in Table 3.12.*

Step 2*: For each artificial scenario, we solve using CPLEX the corresponding LP_{pmtn} model to compute the optimal durations. We construct an optimal permutation of each artificial scenario solution by solving the subsequent preemptive open shop.*

Step 3 *and* ***Step 4****: We use the ratios and the permutation of the potential scenarios (see Table 3.18) to schedule the potential scenarios according to the artificial scenario schedule (see Appendix D). Then, we compute the local performances (see Table 3.19) and the robustness measures of each artificial scenario solution over the set of potential scenarios (see Table 3.20). As we can notice, the ranking of the artificial*

TABLE 3.18: Optimal schedules of the artificial scenarios in the preemptive case: durations (ratios) and permutation

s_{max}	J_1	J_2	J_3	J_4	$\prec$
M_1	1.25 (0.06)	4.5 (0.16)	25.0 (1.0)	0.0 (0.0)	$J_1 \prec J_3 \prec J_2$
M_2	18.75 (0.94)	0.0 (0.0)	0.0 (0.0)	12.5 (0.25)	$J_4 \prec J_1$
M_3	0.0 (0.0)	12.50 (0.83)	0.0 (0.0)	18.75 (0.75)	$J_2 \prec J_4$
$s_{average}$	J_1	J_2	J_3	J_4	$\prec$
M_1	1.87 (0.13)	2.50 (0.11)	18.33 (1.0)	0.0 (0.0)	$J_1 \prec J_3 \prec J_2$
M_2	12.22 (0.87)	0.0 (0.0)	0.0 (0.0)	10.45 (0.30)	$J_4 \prec J_1$
M_3	0.0 (0.0)	10.41 (0.89)	0.0 (0.0)	12.29 (0.70)	$J_2 \prec J_4$
s_{median}	J_1	J_2	J_3	J_4	$\prec$
M_1	1.66(0.12)	1.66 (0.07)	20.0 (1.0)	0.0 (0.0)	$J_2 \prec J_3 \prec J_1$
M_2	11.66 (0.88)	0.0 (0.0)	0.0 (0.0)	11.66 (0.33)	$J_1 \prec J_4$
M_3	0.0 (0.0)	11.66 (0.93)	0.0 (0.0)	11.66 (0.67)	$J_4 \prec J_2$
s_{min}	J_1	J_2	J_3	J_4	$\prec$
M_1	0.0 (0.0)	7.35 (0.50)	7.06 (0.94)	0.0 (0.0)	$J_2 \prec J_3$
M_2	10.0 (1.0)	0.0 (0.0)	0.59 (0.06)	3.82 (0.15)	$J_3 \prec J_1 \prec J_4$
M_3	0.0 (0.0)	3.82 (0.50)	0.0 (0.0)	10.59 (0.84)	$J_2 \prec J_4$
s_{random}	J_1	J_2	J_3	J_4	$\prec$
M_1	0.0 (0.0)	0.0 (0.0)	25.0 (1.0)	0.0 (0.0)	J_3
M_2	13.33 (1.0)	0.0 (0.0)	0.0 (0.00)	5.0 (0.2)	$J_1 \prec J_4$
M_3	0.0 (0.0)	15.0 (1)	0.0 (0.0)	10.0 (0.8)	$J_4 \prec J_2$

TABLE 3.19: Local performances of the artificial scenario solutions in the splitting case

	Measures	s_1	s_2	s_3	s_4	s_5	s_6
X_{max}	Makespan	31.25	23.5	30	22	30.83	27.5
	Regret	6.25	3.5	4.12	2	5.83	2.5
$X_{average}$	Makespan	32.5	23.81	29.44	22.9	30	27.9
	Regret	7.5	3.81	3.56	2.9	5	2.9
X_{median}	Makespan	33.3	25	28.33	23.33	28.66	31.66
	Regret	8.3	5	2.45	3.33	3.66	6.66
X_{min}	Makespan	35.19	27.73	37.7	25.75	40.2	32.65
	Regret	10.19	7.73	11.82	5.75	15.2	7.65
X_{random}	Makespan	32.5	21.66	28.5	24	25	27
	Regret	7.5	1.66	2.62	4	0	2

scenario solutions according to the robustness measures shows that for the decision-maker who aims to

TABLE 3.20: Robustness measures of the artificial scenario solutions in the preemptive case

Artificial scenario solution	Worst-case makespan	Maximal regret
X_{max}	31.25	6.25
$X_{average}$	32.5	7.5
X_{median}	33.3	8.3
X_{min}	40.02	10.19
X_{random}	32.5	7.5

TABLE 3.21: Local performances of the nominal scenario solution in the preemptive case

X_{s1}	s_1	s_2	s_3	s_4	s_5	s_6
Makespan	25	43.33	46.67	28.34	46.67	46.67
Regret	0	23.33	20.78	8.34	21.67	21.67

minimize the worst-case makespan and the maximal regret, the solution to choose is the artificial scenario s_{max} solution. We notice a good improvement compared to the robustness measures of $s_{nominal}$. In fact, the worst-case makespan of the nominal scenario is equal to 46.8 units (see Table 3.21).

3.4 Computational results

3.4.1 Data generator and test protocol

In the numerical experiments, we consider different problem sizes (Table 3.22):

TABLE 3.22: Instance size generation

	small size	medium size	high size
low variety	(m=3,n=10)	(m=7,n=10)	(m=15,n=15)
medium variety	(m=3,n=50)	(m=7,n=50)	(m=15,n=50)
high variety	(m=3,n=200)	(m=7,n=200)	(m=15,n=200)

For each instance, a nominal scenario of processing times p_{ij}^n is randomly generated using a uniform distribution in the interval I=$[5, 50]$. Around this nominal scenario, we generate the potential scenarios according to the following formula:

$$p_{ij}^s = p_{ij}^n(1 + \zeta_{ij}) \; \forall s, \forall i, \forall j \tag{3.35}$$

Where ζ_{ij} is a random variable that reflects the variation. According to ζ_{ij} we distinguish three uncertainty degree:

- for $d = 1$ we have a low uncertainty degree where the potential scenarios vary between -10% and 10% around the nominal scenario: $\zeta_{ij} \sim \mathcal{U}_{[-0.1,0.1]}$,
- for $d = 2$ we have a medium uncertainty degree where the potential scenarios vary between -50% and 50% around the nominal scenario: $\zeta_{ij} \sim \mathcal{U}_{[-0.5,0.5]}$,
- and for $d = 3$ we have a high uncertainty degree where the potential scenarios vary between -100% and 100% around the nominal scenario: $\zeta_{ij} \sim \mathcal{U}_{[-1,1]}$.

The artificial scenario based approach algorithm is coded in Java using $ILOG\ CPLEX\ Concert$ to solve the mathematical formulations LP_{Split} and LP_{pmtn}. We have generated 1000 replications of nominal scenarios for each combination of m, n. We have varied the number of scenarios around the nominal one from 1 to 100 scenarios. Over the computational results, we use as robustness cost indicator the worst-case makespan deviation from the nominal makespan (Equation 3.36). This indicator allows to measure the rate of increase in the makespan when we opt for a robust solution X instead of a classical one, and it is computed below. $C_{max}^{s_n^*}$ is the optimal makespan of a nominal instance when we suppose that the problem is not affected by uncertainty of processing times.

$$\frac{\max_{s\in\Omega_k}\{C_{max}^s(X)\} - C_{max}^{s_n^*}}{C_{max}^{s_n^*}}, \tag{3.36}$$

In all the reported figures, the horizontal axis represents the number of potential scenarios and the vertical axis represents the mean values of the worst-case makespan robustness cost (the standard deviations are

negligible). Each curve is relative to an artificial scenario solution. We report the results according to the three uncertainty degrees ($d = 1, 2, 3$). We distinguish the case of splitting and the case of preemption.

3.4.2 Robustness cost of the artificial scenario solution evaluation

Artificial scenario solution robustness cost evaluation in the case of splitting

In the case of splitting, ***under low uncertainty degree***, the results show that the worst-case makespan robustness cost of all the artificial scenario solutions follow a logarithmic trend line (*e.g.* Figure 3.7 and Figure 3.8). For all the curves, the R-squared value are superior to 0.95, which is a good fit of the line to the data.

The analytic equation of the logarithmic trend line is given as:

$$y = b.ln(x) + b_0 \tag{3.37}$$

where y is the worst-case makespan robustness cost and x is the number of potential scenarios considered besides the nominal one. The interpretation of the coefficients is as follows. When we add the first potential scenario to the nominal one ($x = 1$), the worst-case makespan robustness cost is equal to b_0. Thus, b_0 can be considered as the initial robustness cost: when we cover a scenario besides the nominal one, the makespan increase by $b_0\%$ compared to the case under which the processing times are certain.

$$\frac{dy}{dx} = \frac{b}{x} \tag{3.38}$$

Thus,

$$b = \frac{dy}{\frac{dx}{x}} \tag{3.39}$$

i.e

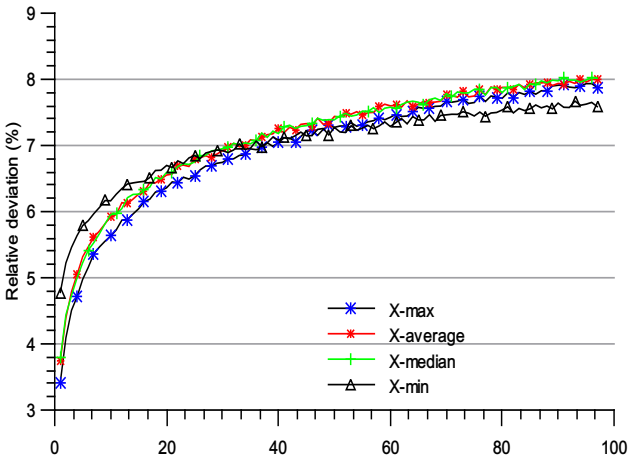
$$\frac{b}{100} = \frac{dy}{\frac{100dx}{x}} \tag{3.40}$$

If the number of scenarios increases with one percent ($\frac{dx}{x} = 1\%$), then the robustness cost will increase with $\frac{b}{100}$. Thus, b can be considered as the increase rate: Each percent change in x is associated with a change of $0.01b$ in y.

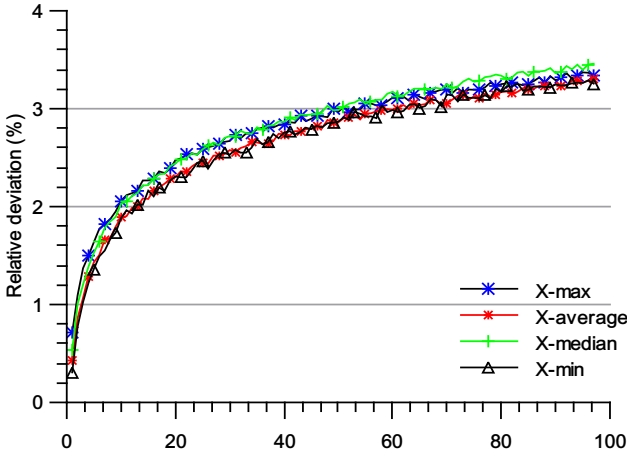
In small size workshops under small variety of jobs (Figure 3.7 graph a. and table d), the robustness costs are comprised between 3.5% and 8%. The logarithmic functions of the artificial solutions ($X_{average}$ and X_{median}) have very similar coefficients: around 3.7% as initial robustness cost and around 0.9% as an increase rate. X_{max} initial robustness cost is slightly inferior to 3.3% but its increase rate is around 1%. X_{min} has the highest initial robustness cost 4.77% but the lowest increase rate 0.62%.

When we increase the number of jobs, both the initial robustness costs and the increase rates decrease. For instance, under medium job variety (Figure 3.7 graphic b. and table d.), the robustness costs are comprised between 0.5% and 3.5%. The initial robustness costs are around 0.5% and the maximal rate of increase is equal to 0.63% and under high job variety (Figure 3.7 graph c. and table d.), the robustness costs

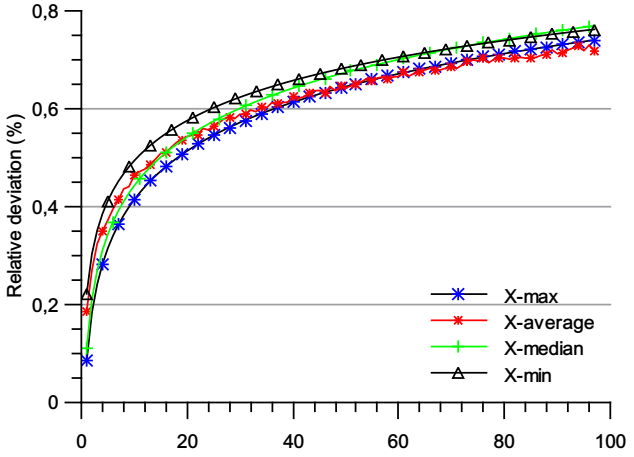
FIGURE 3.7: Artificial scenario solution worst-case makespans in the splitting case: small size workshops under low uncertainty



a. (m=3,n=10,d=1)



b. (m=3,n=50,d=1)



c. (m=3,n=200,d=1)

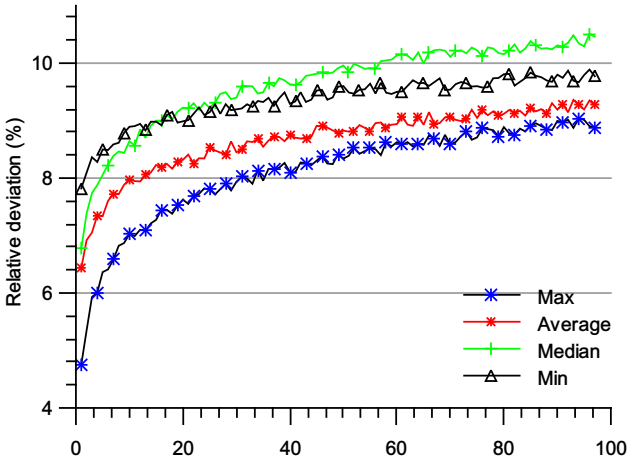
m=3	n	y	R^2
X_{max}	10	0.99ln(x) + 3.37	0.97
	50	0.58ln(x) + 0.49	0.97
	200	0.14ln(x) + 0.11	0.97
$X_{average}$	10	0.93ln(x) + 3.76	0.96
	50	0.62ln(x) + 0.4148	0.96
	200	0.14ln(x) + 0.08	0.98
X_{median}	10	0.93ln(x) + 3.74	0.98
	50	0.63ln(x) + 0.51	0.97
	200	0.11ln(x) + 0.12	0.96
X_{min}	10	0.62ln(x) + 4.77	0.96
	50	0.63ln(x) + 0.57	0.97
	200	0.11ln(x) + 0.12	0.96

d. logarithmic trends

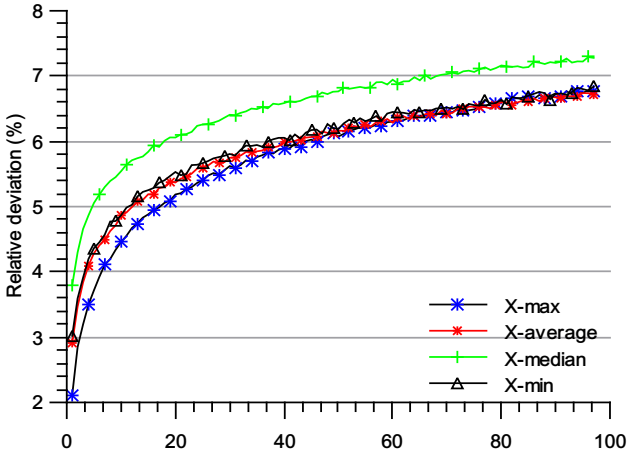
are comprised between 0.12% and 0.8%, the initial robustness costs are around 0.1% and the rate of increase are also around 0.1%.

When we increase the number of machines, the initial robustness costs increase, but the increase is small. For instance, in medium size workshops under small job variety (Figure 3.8 graphic a. and table d.), the robustness costs are comprised between 4.8% and 12%. X_{max} is the most robust while X_{median} is the least robust. The initial robustness cost of X_{max} equals 4.8% and its increase rate is small (0.68%). The initial robustness cost of X_{median} equals 6.8% and its increase rate is small (0.98%). $X_{average}$ robustness cost gets close to X_{max} robustness cost as the number of scenarios increases. The initial robustness cost

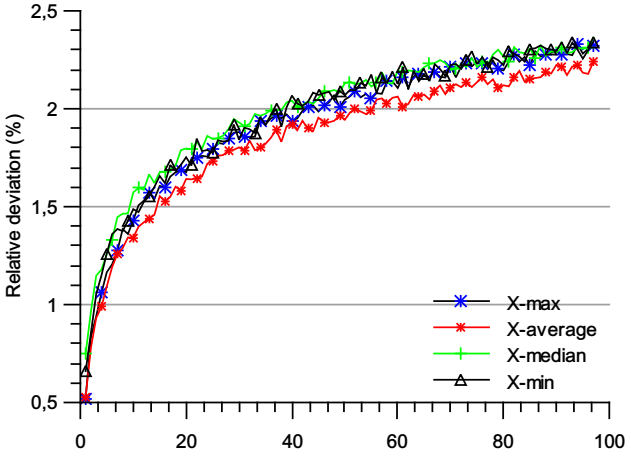
FIGURE 3.8: Artificial scenario solution worst-case makespans in the splitting case: medium size workshops under low uncertainty



a. (m=7,n=10,d=1)



b. (m=7,n=50,d=1)



c. (m=7,n=200,d=1)

m=7	n	y	R^2
X_{max}	10	0.68ln(x) + 4.82	0.97
	50	0.8ln(x) + 2	0.963
	200	0.38ln(x) + 0.54	0.96
$X_{average}$	10	0.60ln(x) + 6.47	0.96
	50	0.53ln(x) + 2.90	0.96
	200	0.37ln(x) + 0.51	0.97
X_{median}	10	0.98ln(x) + 6.80	0.97
	50	0.65ln(x) + 3.80	0.97
	200	0.35ln(x) + 0.72	0.96
X_{min}	10	0.43ln(x) + 7.77	0.96
	50	0.42ln(x) + 3.00	0.99
	200	0.37ln(x) + 0.62	0.98

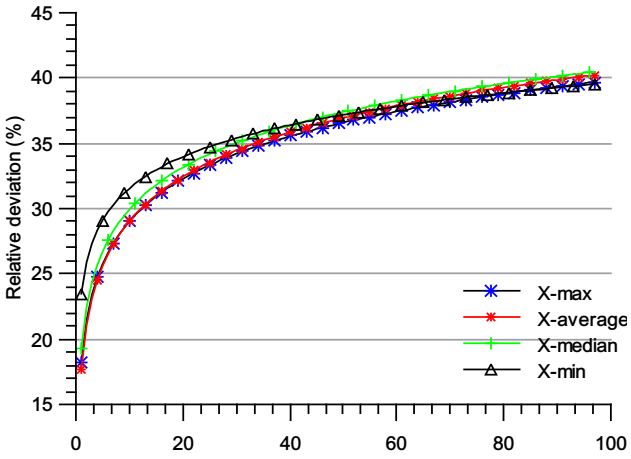
d. logarithmic trends

of X_{min} equals 7.7% and its increase rate is very small (0.43%). When we increase the number of jobs (Figure 3.8 graphics b. c. and table d.), once again both the initial robustness costs and the increase rates of the artificial scenario solutions decrease. The solution ranking according to the robustness cost is the same.

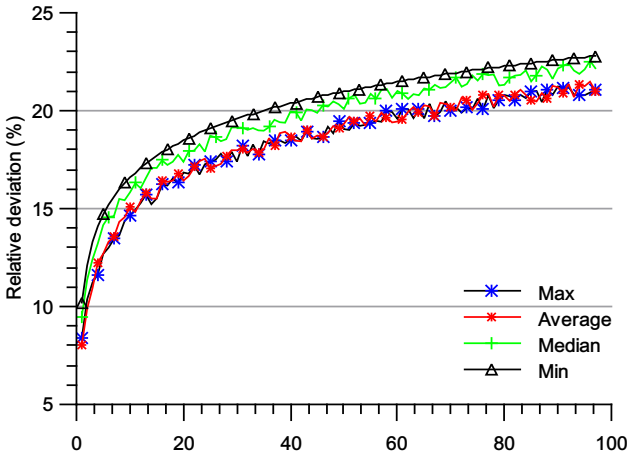
When we increase the degree of uncertainty, the robustness costs highly increase.

Under medium degree of uncertainty, we observe an important increase in which concern both the initial robustness cost and the increase rate of all the artificial scenario solution logarithmic trend lines. X_{min} is clearly the less robust, its robustness cost become more significant as the job variety become important. For instance, in small size workshops under low job variety (Figure 3.9 graphic a. and table d.), the initial

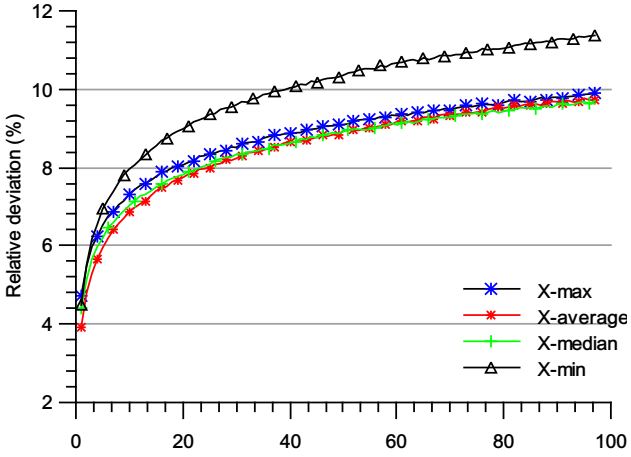
FIGURE 3.9: Artificial scenario solution worst-case makespans in the splitting case: small size workshops under medium uncertainty



a. (m=3,n=10,d=2)



b. (m=3,n=50,d=2)



c. (m=3,n=200,d=2)

m=3	n	y	R^2
X_{max}	10	4.69ln(x) + 18.22	0.98
	50	2.85ln(x) + 7.98	0.96
	200	1.12ln(x) + 4.71	0.97
$X_{average}$	10	4.93ln(x) + 17.68	0.95
	50	2.85ln(x) + 8.03	0.96
	200	1.27ln(x) + 3.90	0.97
X_{median}	10	4.63ln(x) + 19.26	0.96
	50	2.86ln(x) + 9.07	0.97
	200	1.16ln(x) + 4.35	0.96
X_{min}	10	3.51ln(x) + 23.45	0.96
	50	2.74ln(x) + 10.22	0.97
	200	1.50ln(x) + 4.49	0.98

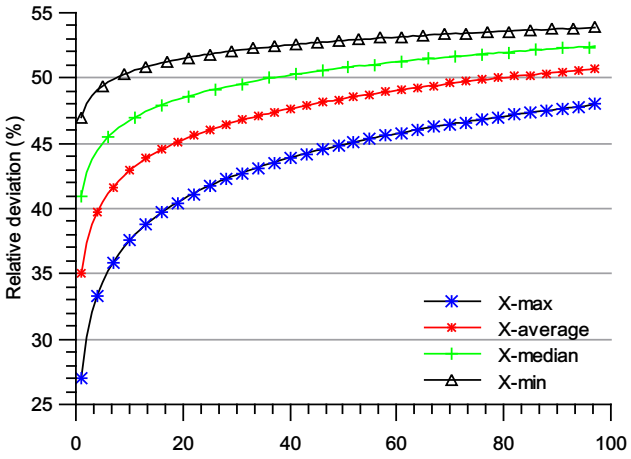
d. logarithmic trends

robustness cost of X_{min} is equal to 23% which represents a huge increase compared with the case of low uncertainty degree (more than five times more expensive) and the increase rate is equal to 3.5%. X_{max}, $X_{average}$ and X_{median} solutions have closest trend lines. Their initial robustness cost is around 18.22% (more than five times more expensive) and their increase rate is around 5%.

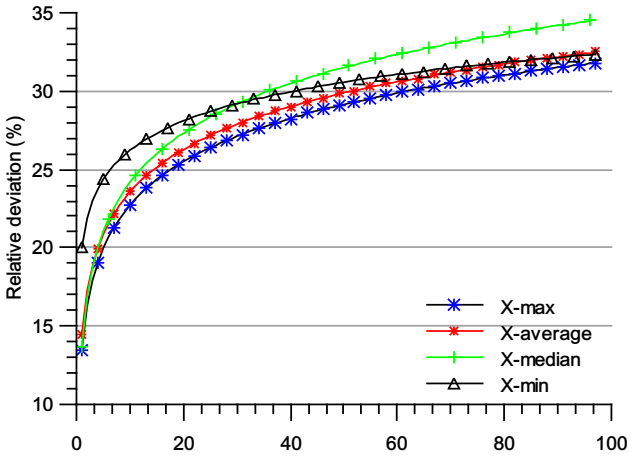
The impact of the number of machines and the number of jobs increasing is still the same (*e.g.* Figure 3.9 graphic b. and Figure 3.10).

Under high uncertainty degree, the robustness cost of X_{max}***,*** $X_{average}$ ***and*** X_{median} ***are still following logarithmic trend lines***. The increase in both the initial robustness cost and the increase rate of X_{max},

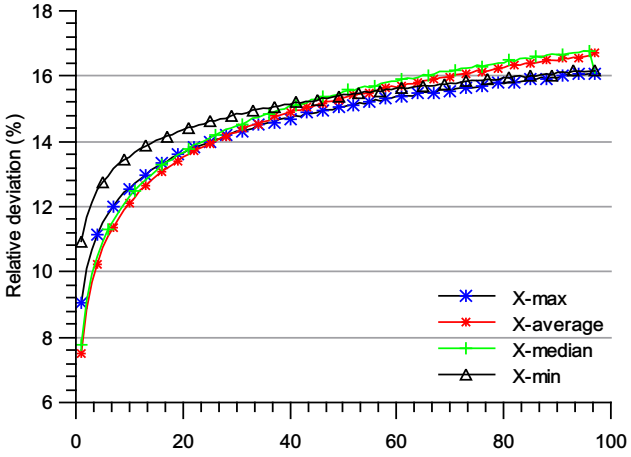
FIGURE 3.10: Artificial scenario solution worst-case makespans under splitting: medium size workshops under medium uncertainty



a. (m=7,n=10,d=2)



b. (m=7,n=50,d=2)



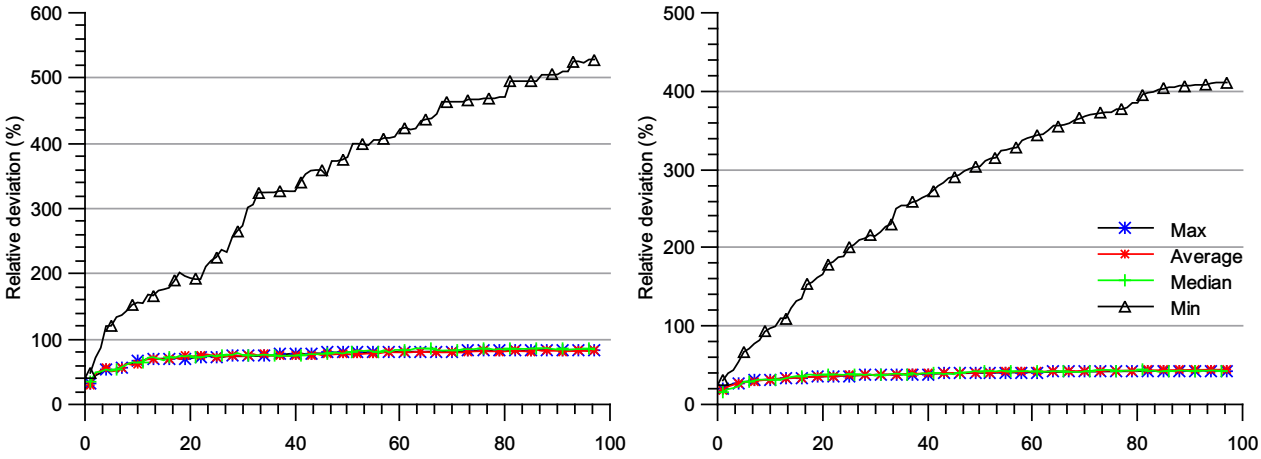
c. (m=7,n=200,d=2)

m=7	n	y	R^2
X_{max}	10	4.58ln(x) + 27.00	0.96
	50	3.99ln(x) + 13.51	0.97
	200	1.54ln(x) + 9.01	0.97
$X_{average}$	10	3.43ln(x) + 34.98	0.96
	50	3.93ln(x) + 14.51	0.96
	200	2.00ln(x) + 7.49	0.98
X_{median}	10	2.50ln(x) + 41	0.96
	50	4.55ln(x) + 13.69	0.98
	200	1.97ln(x) + 7.76	0.97
X_{min}	10	1.50ln(x) + 44	0.95
	50	2.99ln(x) + 22.01	0.97
	200	1.97ln(x) + 11.01	0.96

d. logarithmic trends

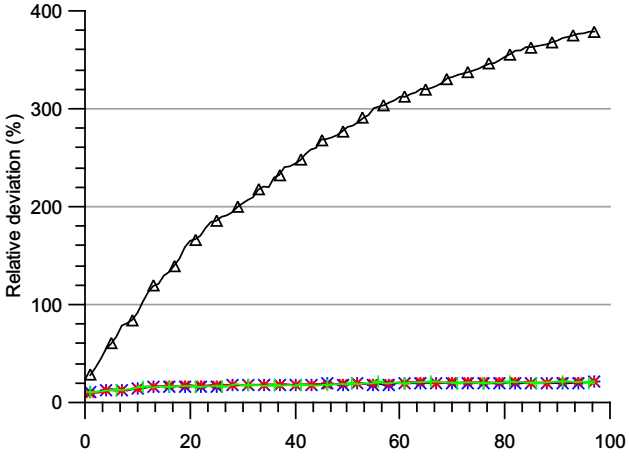
$X_{average}$ and X_{median} is huge. For instance, in small size workshops under small job variety (Figure 3.11 graphic a. and table d.), the initial robustness cost of X_{max}, X_{median} and $X_{average}$ is around 40% (the robustness cost is 10 times more expensive compared to low uncertainty). The increase rates have also increased: around 10% for X_{max} and X_{median} and around 9.2% for $X_{average}$.

FIGURE 3.11: Artificial scenario solution worst-case makespans under splitting: small size workshops under high uncertainty



a. (m=3,n=10,d=3)

b. (m=3,n=50,d=3)



c. (m=3,n=200,d=3)

m=3	n	y	R^2
X_{max}	10	9.86ln(x) + 40.27	0.97
	50	5.59ln(x) + 18.89	0.97
	200	2.46ln(x) + 8.15	0.97
$X_{average}$	10	9.27ln(x) + 41.60	0.96
	50	4.94ln(x) + 20.78	0.97
	200	2.86ln(x) + 8.96	0.96
X_{median}	10	10.21ln(x) + 40.28	0.97
	50	5.64ln(x) + 18.78	0.96
	200	2.75ln(x) + 8.33	0.97
X_{min}	10	-0.03x^2 + 7.8x+69.1	0.99
	50	-0.03x^2 + 7.3x + 29.6	0.96
	200	-0.03x^2 + 6.4x + 35.07	0.97

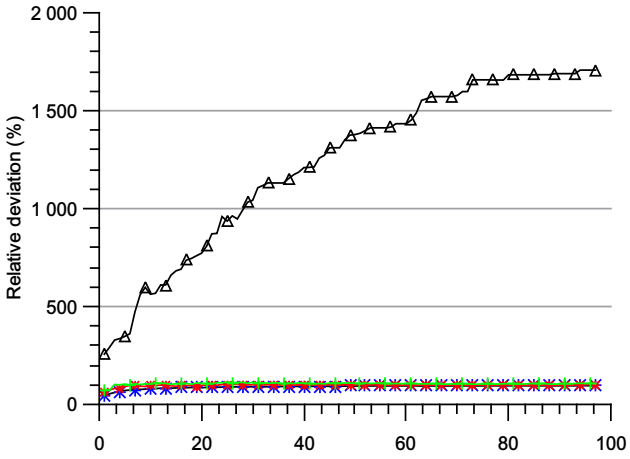
d. logarithmic trends

Under high uncertainty degree particularly, the robustness cost of X_{min} follows a polynomial trend line instead of logarithmic trend line in the tested interval of scenario numbers, we approximate its trend line by a second degree polynomial function when $x \in [1, 100]$:

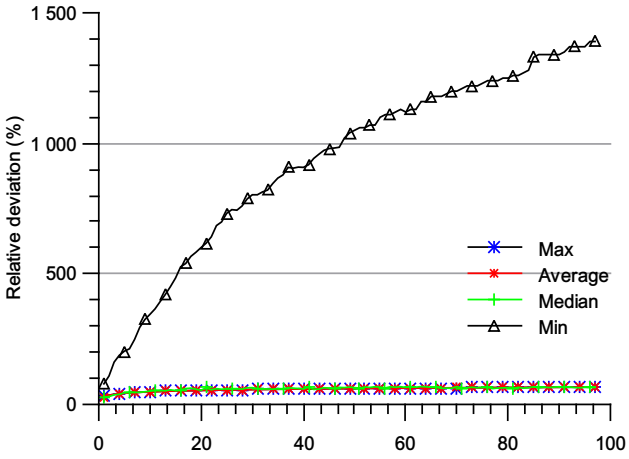
$$y = a_2x^2 + a_1x + a_0 \quad i.e. \quad \frac{dy}{dx} = 2a_2x + a_1 \tag{3.41}$$

This means the variation of the robustness cost y is linear according to the number of scenarios x. This

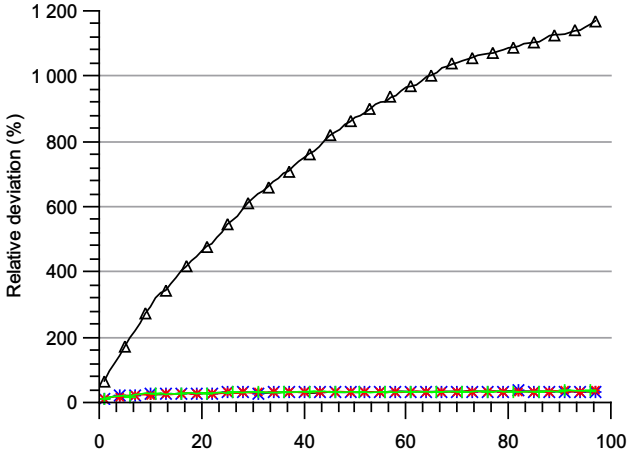
FIGURE 3.12: Artificial scenario solution worst-case makespans under splitting: medium size workshops under high uncertainty



a. (m=7,n=10,d=3)



a. (m=7,n=50,d=3)



a. (m=7,n=200,d=3)

m=7	n	y	R^2
X_{max}	10	9.43ln(x) + 55.75	0.97
	50	7.87ln(x) + 30.14	0.98
	200	4.62ln(x) + 14.05	0.96
$X_{average}$	10	4.23ln(x) + 82.39	0.96
	50	7.97ln(x) + 29.83	0.99
	200	4.58ln(x) + 15.89	0.96
X_{median}	10	2.78ln(x) + 95.70	0.96
	50	7.76ln(x) + 32.47	0.97
	200	4.32ln(x) + 15.28	0.96
X_{min}	10	$-0.1x^2$ + 30.1x + 259.4	0.99
	50	$-0.1x^2$ + 24.7x + 127.0	0.96
	200	$-0.1x^2$ + 20.9x + 80.6	0.96

d. logarithmic trends

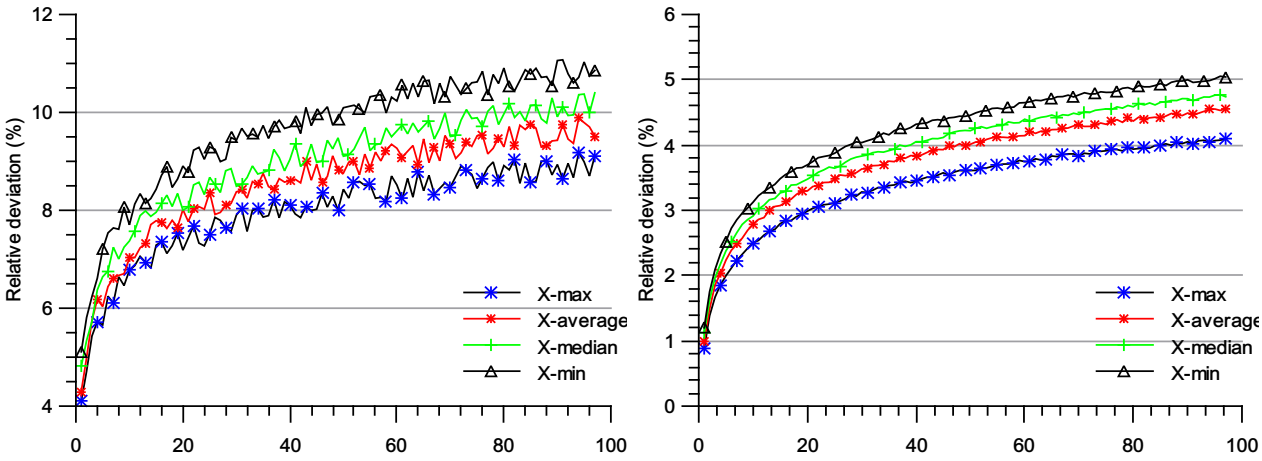
tendency could be explained by the fact that when the number of scenarios increases, the minimal processing times over the scenarios does not increase, in contrary they are more subject to decrease. Under high uncertainty degree, as the number of scenarios increases, the difference between s_{min} and the worst-case scenario that leads the worst-case makespan will be important. Consequently, and contrarily to X_{max} which the solution of the upper extreme scenario, the robustness of X_{min} will highly decrease and consequently, its robustness cost will increase. The trend line increase will continue until we reach the minimum scenario and cover the worst-case scenario over all the potential scenarios that could be generated.

Under high uncertainty degree, we also notice that when we increase the number of jobs, the robustness

costs of the artificial scenario solutions decrease slightly. However, they increase very significantly as the number of machines increases (Figure 3.12): the increasing tendencies are more accurate under high uncertainty than the decreasing tendencies. The computational tests in high size workshops lead to the same conclusions (see Appendix E).

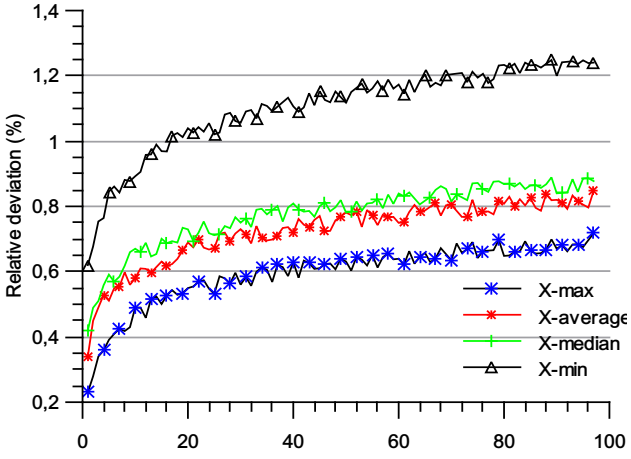
Artificial scenario solution robustness cost evaluation in the case of preemption

FIGURE 3.13: Artificial scenario solution worst-case makespans in the preemptive case: small size workshops under low uncertainty



a. (m=3,n=10,d=1)

b. (m=3,n=50,d=1)



c. (m=3,n=200,d=1)

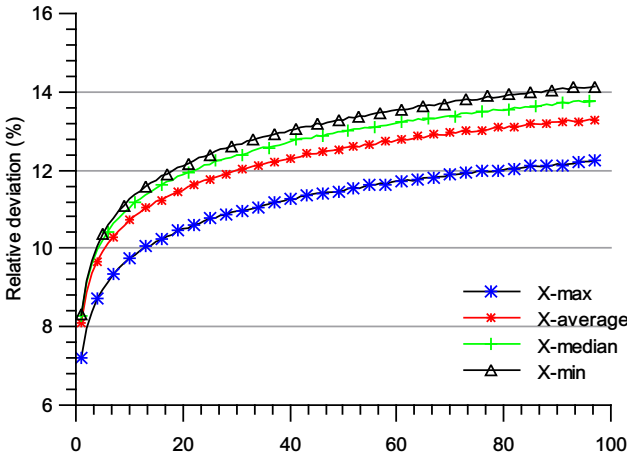
m=3	n	y	R^2
X_{max}	10	1.02ln(x)+4.2	0.97
	50	0.69ln(x) + 0.88	0.97
	200	0.16ln(x) + 0.24	0.975
$X_{average}$	10	1.16ln(x) + 4.35	0.954
	50	0.77ln(x) + 0.98	0.966
	200	0.18ln(x) + 0.37	0.958
X_{median}	10	1.19ln(x) + 4.65	0.968
	50	0.80ln(x) + 1.09	0.973
	200	0.17ln(x) + 0.42	0.956
X_{min}	10	1.22ln(x) + 5	0.968
	50	0.83ln(x) + 1.19	0.973
	200	0.15ln(x) + 0.62	0.956

d. logarithmic trends

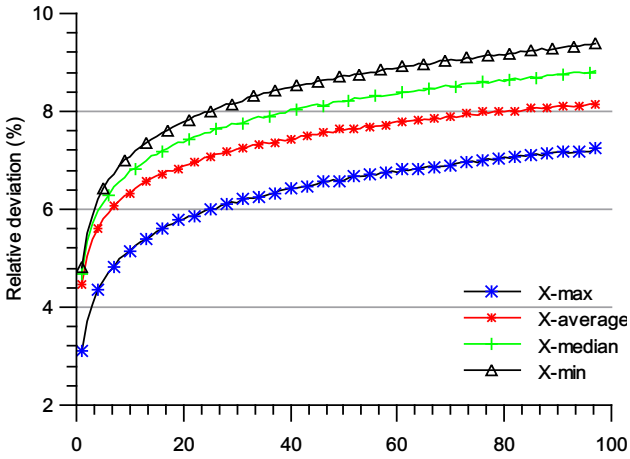
In this section, we report the computational results in the preemptive case. We remark that the preemptive case shares common properties with the splitting case concerning the evolution of the robustness costs

according to the number of scenarios, the number of jobs, the number of machines and the uncertainty degrees. The robustness costs of the artificial scenario solutions also follow a logarithmic trend line under low and medium uncertainty degrees (see Figure 3.13, Figure 3.14).

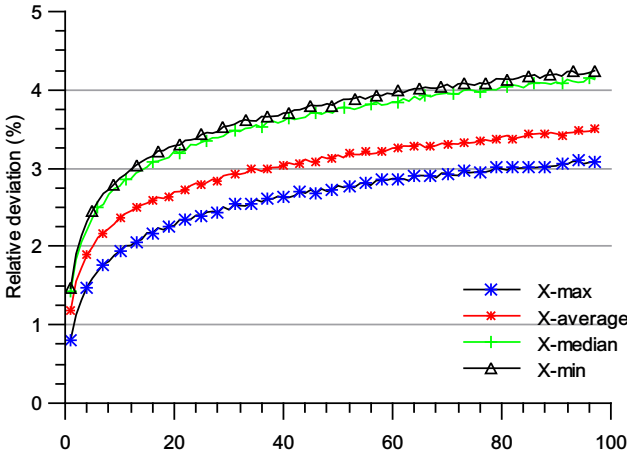
FIGURE 3.14: Artificial scenario solution worst-case makespans in the preemptive case: medium size workshops under low uncertainty



a. (m=7,n=10,d=1)



b. (m=7,n=50,d=1)



c. (m=7,n=200,d=1)

m=7	n	y	R^2
X_{max}	10	1.11ln(x) + 7.2	0.967
	50	0.92ln(x) + 3.14	0.963
	200	0.51ln(x) + 0.81	0.969
$X_{average}$	10	1.14ln(x) + 8.14	0.964
	50	0.83ln(x) + 4.59	0.962
	200	0.53ln(x) + 1.25	0.974
X_{median}	10	1.25ln(x) + 8.3	0.954
	50	1.2ln(x) + 4.76	0.975
	200	0.60ln(x) + 1.52	0.966
X_{min}	10	1.63ln(x) + 9.1	0.962
	50	1.28ln(x) + 4.82	0.979
	200	0.62ln(x) + 1.5	0.968

d. logarithmic trends

The robustness costs increase as the number of machines increases and decrease as the number of jobs increases. But we notice a major difference compared to the splitting case: overall, X_{max} is the most robust and X_{min} is the less robust. Moreover, the robustness costs are slightly more important compared to the case of splitting. For instances:

For instance, ***under low uncertainty degree*** in small size workshops under small job variety (Figure 3.13

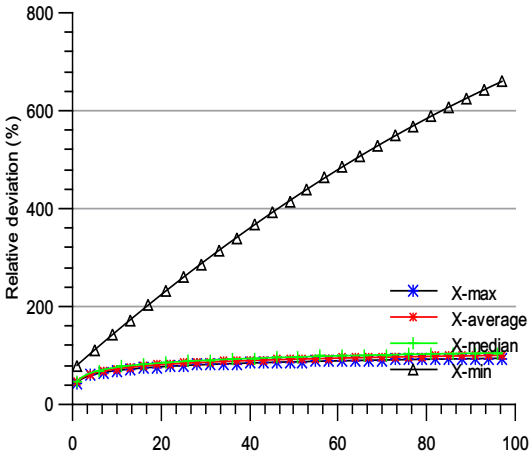
graphic a. and table d.), the robustness costs of the artificial scenario solutions are comprised between 4.2% and 11%. The initial robustness cost of X_{max} which is the most robust is equal to 4.2% and its increase rate is equal to 1%. X_{min} is the less robust, its initial robustness cost is equal to 5% and its increase rate is equal to 1.2%. ***When we increase the number of jobs***, both the initial robustness costs and the increase rates decrease. Under medium job variety Figure 3.13 graphic b. and table d.), the robustness costs of the artificial scenario solutions are comprised between 1.2% and 4.8% *i.e.* 3.5 times less expensive.

When we increase the numbers of machines, we notice an important increase in the robustness cost. And this increase is also significant compared to the splitting case. For instance in medium size workshops under small job variety Figure 3.14 graphic a. and table d.), the robustness costs are comprised between 13% and 23%.

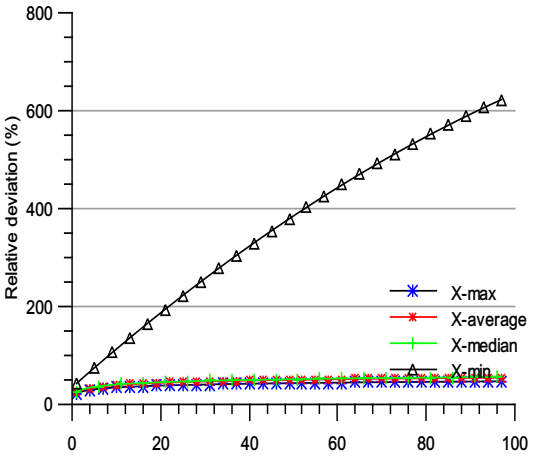
Under medium degree of uncertainty, in small size workshops under low job variety (Figure E.2 graphic a. and table d.), the robustness cost of X_{max} is comprised between 21.18% and 42% while the robustness costs of the other solutions are comprised between 23% and 110%. The initial robustness cost of X_{max} is equal to 21.18% and its increase rate is equal to 15.21%. The initial robustness cost of $X_{average}$ is equal to 23% and its increase rate is equal to 15.60%. The initial robustness cost of X_{median} is equal to 27.18% and its increase rate is equal to 17.25%. The initial robustness cost of X_{min} is equal to 28.45% and its increase rate is equal to 1.69%.

Finally, ***under high uncertainty degree***, the robustness cost of X_{max}, $X_{average}$ and X_{median}, like in the splitting case, still follow a logarithmic trend lines while the robustness cost of X_{min} follow a polynomial trend line exactly like in the case of splitting. For example, in small size workshops under small job variety (Figure 3.15 graphic a. and table d.), the worst-case makespan deviations of X_{max}, X_{median} and $X_{average}$ is between 44% and 53%. The initial robustness cost of X_{max} is equal to 44.31% and its increase rate is equal to 11.23%. The initial robustness cost of $X_{average}$ is equal to 45.56% and its increase rate is equal to 12.57% and the initial robustness cost of X_{median} is equal to 53% and its increase rate is equal to 13.24%. When we increase the number of job, the robustness cost decreases. Under medium job variety (Figure 3.15 graphic b. and table d.), the initial robustness cost of X_{max} and $X_{average}$ is around 24% while the initial robustness cost of X_{median} is equal to 26.45%. The increase rate is equal to 11.43% for X_{max}, 12.13% for $X_{average}$, and 13.45% for X_{median}. Under high job variety (Figure 3.15 graphic c. and table d.), the initial robustness cost of X_{max} is around 12% while the initial robustness cost of $X_{average}$ and X_{median} is around 14%. The increase rate is equal to 5.5% for X_{max} and 6.73% for $X_{average}$, and its around 6.85% for X_{median}. The computational tests in high size workshops lead to the same conclusions (see Appendix E).

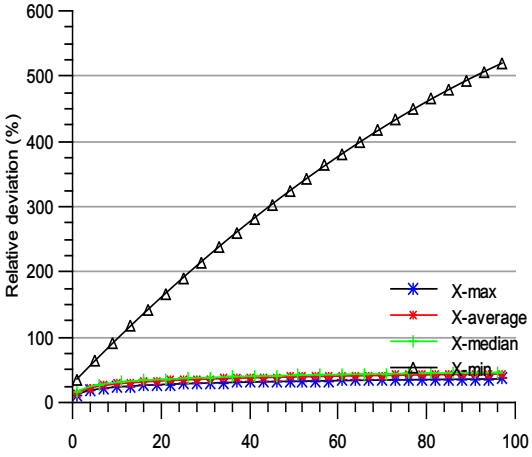
FIGURE 3.15: Artificial scenario solution worst-case makespans in the preemptive case: small size workshops under high uncertainty



a. (m=3,n=10,d=3)



b. (m=3,n=50,d=3)



c. (m=3,n=200,d=3)

m=3	n	y	R^2
X_{max}	10	11.23ln(x) + 44.31	0.978
	50	11.43ln(x) + 24.05	0.978
	200	5.5ln(x) +13.2	0.973
$X_{average}$	10	12.57ln(x) + 45.56	0.950
	50	12.13ln(x) + 24.44	0.971
	200	6.73ln(x) +13.08	0.956
X_{median}	10	13.24ln(x) + 53	0.954
	50	13.45ln(x) + 26.45	0.956
	200	6.85ln(x) +13	0.952
X_{min}	10	-0.02x^2 + 8.4x+72	0.994
	50	-0.03x^2 + 8.67x + 35.16	0.956
	200	-0.02x^2 + 7.48x + 29.07	0.952

d. line trend functions

3.4.3 Results discussion and analysis

From the computational results, we can deduce that the robustness cost of the robust artificial scenario solutions increases as the number of scenarios increases or the degree of uncertainty increases. Indeed, when we need to cover more potential scenarios, we need to pay additional cost to be protected against uncertainty. But, as it was shown in the results, most of the robust artificial solutions follow a logarithmic trend line which means that the robustness cost increases quickly when we cover the first potential scenarios besides the nominal one (the initial robustness cost) then for each additional scenario, we pay a marginal cost.

Under small, medium and high uncertainty degree, X_{max}, $X_{average}$ and X_{median} follow logarithmic trend lines. Their robustness costs increase as the degree of uncertainty increases. Both the initial robustness cost and the increase rate are concerned by a such increase and this can be explained by the fact that the scenarios become very different.

In which concerns X_{min}, we observe that this solution also follows logarithmic trend lines under small and medium uncertainty degrees. However, under high uncertainty degree, the trend line of X_{min} is no longer logarithmic but it is polynomial for the considered interval of scenario numbers. As explained in the previous sections, the polynomial tendency could be explained by the fact that when the number of scenarios increase, the minimal processing times over the scenarios does not increase, in contrary they are more subject to decrease. As the number of scenarios that belongs to high uncertainty degree increase, the difference between s_{min} and the worst-case scenario over the set of scenarios becomes important. Consequently, the ability of X_{min} to withstand the worst-case scenario is small because when constructing s_{min}, we do not provide any temporal margin. Consequently, X_{min} do not absorb the variations which lead to a robustness cost that highly increases.

Furthermore, we observed that the robustness cost of the solutions increases as the number of machines increases and decreases as the number of jobs increases. This tendency can be related to the performance of the nominal solution more than the performance of the robust solutions. Indeed, when the number of machines is important, the optimization in the deterministic case of the nominal scenario can lead to a small makespan value which makes the deviation between this one and the worst-case makespan very important and consequently, it increases the robustness cost. But when the number of jobs is important while the number of machines is small, the optimization in the deterministic case of the nominal scenario can lead to an important makespan value which make the deviation between this one and the worst-case makespan very small and consequently it decreases the robustness cost which makes the robust solutions more interesting.

In the case of splitting under small and medium uncertainty degrees, the robust scenario solutions X_{max}, $X_{average}$, X_{median} and X_{min} lead to similar robustness costs except when the number of jobs is close to the number of machines. Under this special case, X_{max} solution is clearly more robust. And under high uncertainty degree, the robustness cost of X_{min} is extremely high.

Contrarily to the splitting case, in the preemptive case, the artificial scenario solution X_{max} is more robust than the other artificial scenario solutions, followed by $X_{average}$ and X_{median}, while X_{min} solution is the least robust. This ranking is the same for all problem sizes and under all the uncertainty degrees. Besides, the artificial scenario solutions follow the same trend lines as in the splitting case but their initial robustness costs and their increase rates are more important. This increase is due to the non-overlapping constraints that induces idle times under the preemptive case. Moreover, the non-preemptive constraints allow to X_{max}, which offers temporal protection, to perform better than the other robust solutions.

Conclusion

In this chapter, we have considered the makespan minimization on parallel machines under both splitting and preemption under uncertain processing times. We have shown first that the optimal solutions computed

based on the nominal scenario leads to infeasible schedules, or schedules with poor performance when the processing times are uncertain. We then proposed a preliminary approach that aims to enforce the feasibility of the schedule under different scenarios by tolerating the violations of some processing requirements. Slack variables are used to compensate a such a loss. However, the results show that under this approach the overproduction or the shortage are very important. Consequently, additional costs are needed to make the adjustments for all the scenarios. To avoid such efforts, we have proposed an artificial scenario based approach that does not require the slacks.

Under this approach, we have constructed a set of feasible solutions based on a set of artificial scenarios. Then, we have chosen the robust solutions based on an evaluation algorithm that compute a robustness measure of each solution. The approach is resumed on a general algorithm composed of 4 steps with specific instructions in the case of preemption (to take into account the sequence decision). Thanks to this approach, the hard constraints are respected and the feasibility is insured without recourse actions.

We have done extensive computational experiments to test the validity of such an approach. Besides, we have shown that in both splitting and preemption case, the robustness cost of the artificial scenario solutions follow logarithmic line-trends according to the number of scenarios under low and medium uncertainty degrees. Under high uncertainty degree, the robustness cost of the considered artificial scenario solutions follow logarithmic line-trends except s_{min} solution that has a polynomial line-trend which is very huge. Nevertheless, it will be necessary to verify this logarithmic tendency on other data generated differently to be sure that it does not depend on the data generation. Moreover, the interpretation of the solution X_{min} polynomial trend line should be questioned especially that the number of considered scenarios in the computational test was limited to 100 scenarios.

In the next chapter, we will use the robust discrete optimization framework developed by Kouvelis and Yu (1997) to compute the optimal robust solutions for the makespan minimization on unrelated parallel machines with job splitting (*resp.*. preemption) under uncertain processing times. The objective is to reduce more the robustness cost of the solutions.

Chapter 4

Robust parallel machine scheduling with job splitting under uncertain processing times

Abstract: The goal of this chapter is to propose robust formulations, based on the robust discrete optimization framework developed by Kouvelis and Yu (1997), to compute the optimal robust solutions for the makespan minimization on unrelated parallel machines with job splitting under uncertain processing times. We show that the direct application of robust optimization techniques leads to no solution. To overcome this difficulty, we propose a formulation in which the uncertainty appears in the objective instead of the assignment constraints. Accordingly, we propose two formulations to compute robust solutions with min-max and min-max regret objectives. The computational tests show that these formulations are able to provide robust solutions in polynomial times. Extensive numerical experiments show that these formulations improve the robustness cost compared with the artificial scenario solutions. Lastly, we extend the robust formulations to the special cases of uniform and identical parallel machines.

Contents

Introduction

In this chapter, we apply the robust discrete optimization framework developed by Kouvelis and Yu (1997) to compute optimal robust schedules under uncertain processing times. The robust discrete optimization framework is a comprehensive mathematical programming framework for robust decision making under which the uncertainty is represented by discrete scenarios and the objective is to optimize a robustness measure. Accordingly, the deterministic linear formulation is extended to incorporate a discrete scenario representation and the objective is to optimize a global performance. Under the robust discrete optimization, the constraints are hard *i.e.*,, they must be satisfied for all the scenarios.

In Chapter 3, we have shown that in the formulation of the problem $Rm|Split|C_{max}$, the uncertainty of processing times is affecting the data in the equality constraints while the objective and the inequality constraints are not affected with the uncertainty. Thus, the robust counterpart, which is an extension of the deterministic linear formulation, is a constraint feasibility problem. In this constraint feasibility problem, the objective is to find the schedule that is feasible under any scenario as expressed in the constraints below:

$$\sum_{i=1}^{m} \frac{t_{ij}}{p_{ij}^{s}} = 1, \quad j = 1, \ldots, n, \; s = s_1, \ldots, s_k. \tag{4.1}$$

But, the equality constraints cannot be insured for different scenarios. In order to construct a feasible schedule under all the scenarios, we propose to change the decision variables in the deterministic problem. We will consider x_{ij} as the assignment ratio of job j to machine i: x_{ij}=$\frac{t_{ij}}{p_{ij}}$. This transformation will lead to the deterministic linear programming (LP_{assign}) below:

Minimize C_{max}

Subject to

$$\begin{aligned}
\sum_{i=1}^{m} x_{ij} &= 1, && j = 1, \ldots, n, \; s = s_1, \ldots, s_k \\
\sum_{j=1}^{n} p_{ij} x_{ij} - C_{max} &\leq 0, && i = 1, \ldots, m \\
0 \leq x_{ij} &\leq 1, && i = 1, \ldots, m \; j = 1, \ldots, n
\end{aligned}$$

LP_{assign} will return assignment ratios as decision variables instead of temporal durations. Accordingly, the processing times appear in the inequality constraints instead of the equality constraints. Based on this remark, we will be able to formulate and solve the robust versions according to the robust discrete optimization framework.

Let us denote M the set of m machines, N the set of n jobs, S the set of k discrete scenarios, and F the set of all feasible solutions under the constraints of the deterministic LP_{assign} *i.e*

$$F = \{(x_{ij})_{i \in M, j \in N} \in [0,1]^{mn} / \sum_{i=1}^{m} x_{ij} = 1, \forall j \in N\}.$$

To compute optimal robust solutions, we consider: the worst-case makespan which is already presented in Chapter 3 and the maximal regret.

We define the optimal robust solutions as follows:

Definition 4.0.1 *We call a solution X optimal robust with respect to the worst-case makespan if it satisfies the following conditions:*

- *X is feasible for each scenario $s \in S$ taken independently.*
- *Whatever the values of the processing times p_{ij}^s, X minimizes the worst-case makespan over all scenarios.*

Definition 4.0.2 *We call a solution X optimal robust with respect to the regret if it satisfies the following conditions:*

- *X is feasible for each scenario $s \in S$ taken independently.*
- *Whatever the values of the processing times p_{ij}^s, X minimizes the maximal deviation from the optimal makespans over all scenarios.*

In Section 4.1, based on the ratio transformation and the robust solution definitions, we formulate and solve the robust versions of $Rm|Split|C_{max}$ under discrete processing times scenarios. In Section 4.1.1, we formulate the mathematical model that minimizes the worst-case makespan (*DmM*$(Rm|Split|C_{max}; p_{ij})$), respectively, we formulate the model that minimizes the maximal regret (*DmMR*$(Rm|Split|C_{max}; p_{ij})$). We use the same example presented in Chapter 3 to illustrate the approach. In Section 4.2,, we report the computational results based on the data generator given in Chapter 3. We report the computational times required to solve the proposed formulations under different problem sizes. Besides, we evaluate and compare the optimal robust solutions by measuring their the robustness cost, we compare them also to the artificial scenario solutions. In Section 4.3, we address the robust versions of parallel uniform machines $Qm|Split|C_{max}$: *DmM*$(Qm|Split|C_{max}; p_{ij})$ and *DmMR*$(Qm|Split|C_{max}; p_{ij})$ and extend the results to the robust version of identical parallel machines $Pm|Split|C_{max}$): *DmM*$(Pm|Split|C_{max}; p_{ij})$ and *DmMR*$(Pm|Split|C_{max}; p_{ij})$.

4.1 Formulations of the robust unrelated parallel machine scheduling problem with splitting under discrete processing time scenarios

In this section, we provide mathematical formulations to compute the optimal robust schedules on the unrelated parallel machine under splitting. We formulate the discrete min-max version of $Rm|Split|C_{max}$ under uncertain processing times to compute the robust solution minimizing the worst-case makespan in Section 4.1.1. And we formulate the discrete min-max regret version of $Rm|Split|C_{max}$ under uncertain processing times to compute the robust solution minimizing the maximal regret in Section 4.1.2.

We remind that in the splitting case, an optimal robust assignment solution leads to an optimal robust schedule with the same makespan without any sequencing constraints.

4.1.1 Formulation of discrete min-max version of $Rm|Split|C_{max}$ under discrete processing time scenarios

Let X be an element of F the set of feasible solutions. The completion time of machine i when the solution X is applied to the scenario s is defined as

$$C_i^s(X) = \sum_{j=1}^{n} p_{ij}^s x_{ij},$$

and the makespan is defined as

$$C_{max}^s(X) = \max_{i \in M}\{C_i^s(X)\}.$$

The worst-case makespan of solution X over all the scenarios of S is then :

$$Z_a(X) = \max_{s \in S}\{C_{max}^s(X)\}.$$

The discrete min-max version of $Rm|Split|C_{max}$ under uncertain processing times can be formulated as the problem of finding X^* the optimal robust solution in absolute sense, i.e the solution that minimizes the worst-case makespan among all feasible solutions. It can be stated as

$$\min_{X \in F} \max_{s \in S} \max_{i \in M}\{C_i^s(X)\}.$$

Thus, the discrete min-max version of $Rm|Split|C_{max}$ under uncertain processing times could be formulated as the following mathematical model:

$$\text{Minimize } \max_{s \in S} \max_{i \in M}\{C_i^s(X)\}$$

Subject to

$$\begin{aligned} \sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1, \ldots, n \\ 0 \leq x_{ij} &\leq 1, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n. \end{aligned}$$

This formulation is nonlinear, we linearize $\max_{i \in M}\{C_i^s(X)\}$ by introducing a variable C_{max}^s such that

$$C_i^s(X) \leq C_{max}^s, \quad i = 1, \ldots, m, \quad s = s_1, \ldots, s_k.$$

This lead to the following (still nonlinear) formulation:

Minimize $\max_{s\in S}\{C^s_{max}\}$

Subject to

$$\begin{aligned}\sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1,\dots,n\\ \sum_{j=1}^{n} p^s_{ij}x_{ij} - C^s_{max} &\le 0, \qquad i = 1,\dots,m,\ s = s_1,\dots,s_k\\ 0 \le x_{ij} &\le 1, \qquad i = 1,\dots,m,\ j = 1,\dots,n\end{aligned}$$

We linearize also $\max_{s\in S}\{C^s_{max}\}$ by introducing a variable y such that

$$C^s_{max} \quad \le y, \quad s = s_1,\dots,s_k. \tag{4.2}$$

Hence, we obtain the following formulation:

Minimize y

Subject to

$$\begin{aligned}\sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1,\dots,n\\ C^s_{max} &\le y, \qquad s = s_1,\dots,s_k\\ \sum_{j=1}^{n} p^s_{ij}x_{ij} - C^s_{max} &\le 0, \qquad i = 1,\dots,m,\ s = s_1,\dots,s_k\\ 0 \le x_{ij} &\le 1, \qquad i = 1,\dots,m,\ j = 1,\dots,n.\end{aligned}$$

C^s_{max} is an intermediate variable that could be eliminated, and then discrete min-max version of $Rm|Split|C_{max}$ under uncertain processing times can be solved to optimality by the linear program below $LP_{w-Split}$:

Minimize y

Subject to

$$\begin{aligned}\sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1,\dots,n\\ \sum_{j=1}^{n} p^s_{ij}x_{ij} &\le y, \qquad i = 1,\dots,m,\ s = s_1,\dots,s_k\\ 0 \le x_{ij} &\le 1, \qquad i = 1,\dots,m, j = 1,\dots,n.\end{aligned}$$

The uncertain processing times appear in the inequality constraints instead of equality constraints which makes the model solvable.

The robust discrete min-max version of $(Rm|Split|C_{max})$ under uncertain processing times remains an LP. The proposed mathematical formulation has the same number of decision variables $(mn + 1)$ as the $(Rm|Split|C_{max})$ reformulation and a set of additional constraints proportional to the number of scenarios: $n + m * k$. Hence, and based on Khachian (1979); Karmarkar (1984), we have the following result:

Corollary 4.1.1 *DmM$(Rm|Split|C_{max}; p_{ij})$, the robust version the polynomial-time solvable $Rm|Split|C_{max}$, is also polynomial.*

Let us go back to our example 3.2.1.

Example 4.1.1 *We solve $LP_{w-Split}$, the formulation of the discrete min-max version of $(Rm|Split|C_{max})$, under the six scenarios of processing requirements. The optimal robust solution in the absolute sense X_w is reported in Table 4.1. Under this robust solution: 55% of $J1$ is processed on machine 1, and 45% on machine 2. $J2$ is mainly processed on machine 3, only 5% of its requirement is processed on machine 1. 72% of $J3$ is processed on machine 1, 12% is processed on machine 2 and 16% is processed on machine 3. And finally, 43% of $J4$ is processed on machine 2 and 56.4% is processed on machine 3 and only 0.6% is processed on machine 1. The optimal worst-case makespan is equal to 27.16 time units. We notice that in this example, the worst-case makespan of the optimal robust solution X_w is less expensive compared to the worst-case makespan of the robust artificial scenarios and the worst-case makespan of the nominal solution. The gaps reported in Table 4.2 measure the relative deviation between the worst-case makespan of each solution and the worst-case makespan of X_w.*

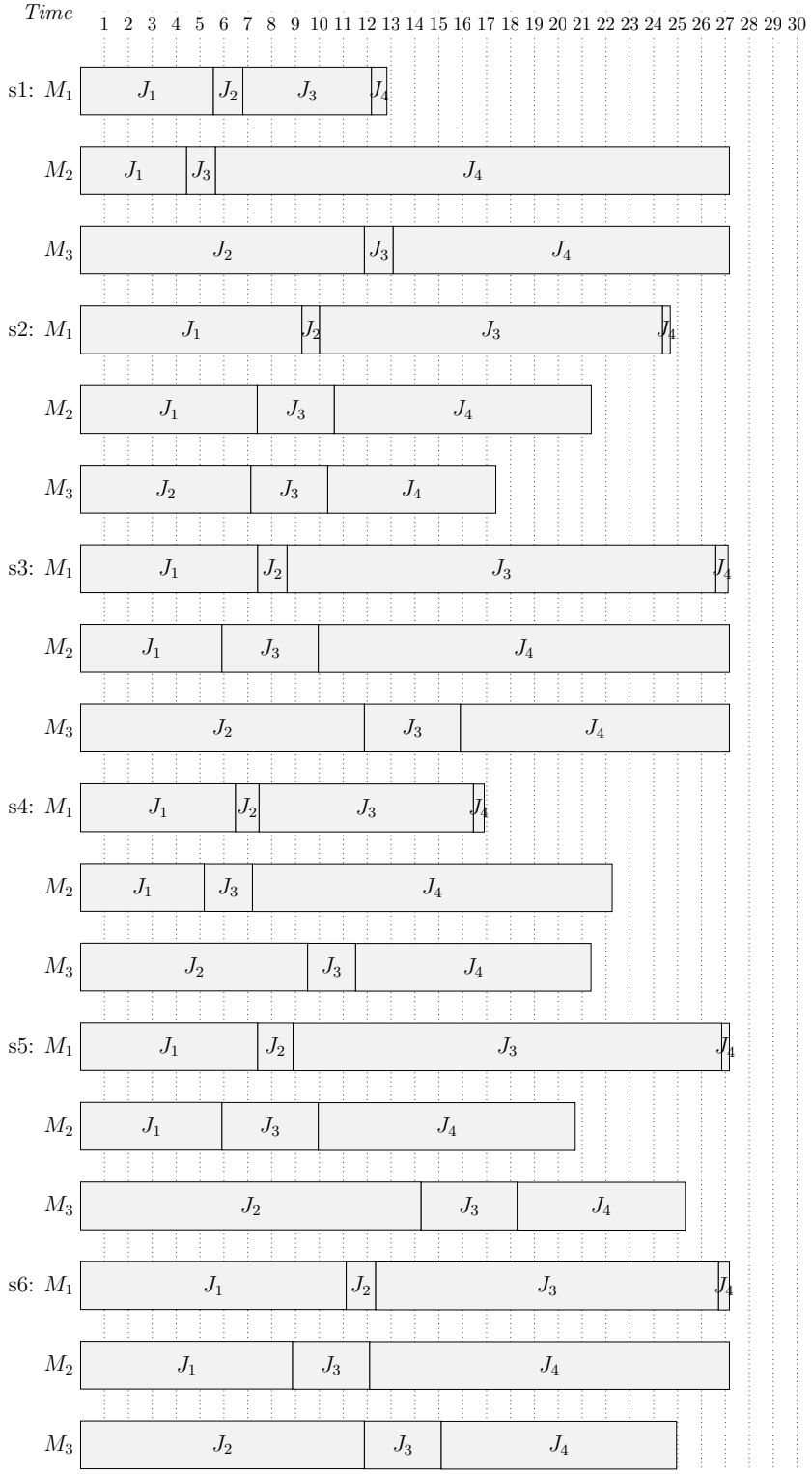
TABLE 4.1: An optimal robust solution minimizing the worst-case makespan in the splitting case: (X_w)

x_{ij}^w	J_1	J_2	J_3	J_4
M_1	0.55	0.05	0.72	0.006
M_2	0.45	0	0.12	0.43
M_3	0	0.95	0.16	0.564

TABLE 4.2: The worst-case makespan gaps under X_w in the splitting case

X	*Worst-case makespan*	Gap(%)
$X_{s_{nominal}}$	45.62	67%
X_{max}	29.16	7.3%
$X_{average}$	29.11	7.17%

FIGURE 4.1: Potential scenarios scheduled according to X_w in the splitting case



4.1.2 Formulation of robust discrete min-max regret version of $Rm|Split|C_{max}$ under discrete processing time scenarios

In this section, we propose the mathematical formulation that computes the optimal robust solution minimizing the regret over all the scenarios.

Let X be an element of F. The maximal deviation of solution X is such that

$$Z_r(X) = \max_{s \in S}\{C^s_{max}(X) - C^{s*}_{max}\}.$$

where C^{s*}_{max} is the optimal makespan of $Rm|Split|C_{max}$ under the scenario s.

The robust discrete min-max regret version of $Rm|Split|C_{max}$ can be formulated as the problem of finding X the optimal robust solution in relative sense, *i.e.*, the solution minimizing the maximal deviation from the optimal makespan over all the scenarios among all feasible solutions $X \in F$. It can be stated as

$$\min_{X \in F} \max_{s \in S}\{\max_{i \in M}\{C^s_i(X)\} - C^{s*}_{max}\}.$$

Thus, the robust discrete min-max regret version of $(Rm|Split|C_{max})$ under uncertain processing times could be formulated as the following mathematical model:

$$\text{Minimize } \max_{s \in S}\{\max_{i \in M} \sum_{j=1}^{n} p^s_{ij} x_{ij} - C^{s*}_{max}\}$$

Subject to

$$\begin{aligned} \sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1, \ldots, n \\ 0 \leq x_{ij} &\leq 1, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n. \end{aligned}$$

This formulation is nonlinear. By the same reasoning as proposed in the previous section, we linearized $\max_{i \in M} \sum_{j=1}^{n} p^s_{ij} x_{ij}$ by introducing a variable z^s

$$\text{Minimize } \max_{s \in S}\{z^s - C^{s*}_{max}\}$$

Subject to

$$\begin{aligned} \sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1, \ldots, n \\ \sum_{j=1}^{n} p^s_{ij} x_{ij} &\leq z^s, \qquad i = 1, \ldots, m,\ s = s_1, \ldots, s_k \\ 0 \leq x_{ij} &\leq 1, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n. \end{aligned}$$

We linearise also $\max_{s \in S}\{z^s - C^{s*}_{max}\}$ by introducing a variable y:

Minimize y

Subject to

$$\begin{aligned}
\sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1, \ldots, n \\
z^s - C^{s*}_{max} &\leq y, \qquad s = s_1, \ldots, s_k \\
\sum_{j=1}^{n} p^s_{ij} x_{ij} &\leq z^s, \qquad i = 1, \ldots, m,\ s = s_1, \ldots, s_k \\
0 \leq x_{ij} &\leq 1, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n.
\end{aligned}$$

This leads to the following linear program:

Minimize y

Subject to

$$\begin{aligned}
\sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1, \ldots, n \\
\sum_{j=1}^{n} {p_{ij}}^s x_{ij} - C^{s*}_{max} &\leq y, \qquad i = 1, \ldots, m,\ s = s_1, \ldots, s_k \\
0 \leq x_{ij} &\leq 1, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n.
\end{aligned}$$

For each scenario s, we must solve a deterministic problem $Rm|Split|C^s_{max}$. Moreover, we can prove that for each scenario s there exists an optimal schedule such that the m machines complete simultaneously. This property can be formulated as

$$C^{s*}_{max} \quad = \sum_{j=1}^{n} p^s_{ij} x^{s*}_{ij}, \quad \forall i \in M, \forall s \in S.$$

Indeed, let us suppose that there exists an optimal schedule in which the m machines do not complete simultaneously. There exists a machine in which the completion time is higher than the completion times of the other machines. If j is the job that increases the completion time, we can split this job j and assign the sub-jobs of j to the other machines. By this procedure, the completion times of jobs are not increasing. We can repeat this procedure until we get an optimal schedule in which the m machines complete simultaneously.

By using this recent property, the new formulation of the problem is ($LP_{r-Split}$):

Minimize y

Subject to

$$\begin{aligned}
\sum_{i=1}^{m} x_{ij} &= 1, \qquad j = 1, \ldots, n \\
\sum_{j=1}^{n} {p_{ij}}^s (x_{ij} - x^{s*}_{ij}) &\leq y, \qquad i = 1, \ldots, m,\ s = s_1, \ldots, s_k \\
0 \leq x_{ij} &\leq 1, \qquad i = 1, \ldots, m, j = 1, \ldots, n.
\end{aligned}$$

Consequently, robust discrete min max regret version of $Rm|Split|C_{max}$ under uncertain processing time can be solved to optimality according to the proposed linear formulation.

The mathematical formulation of the robust discrete min-max regret version of $Rm|Split|C_{max}$ remains a LP as the optimal deterministic solutions of the discrete scenarios can be already calculated by deterministic LPs. Compared to the mathematical model of $Rm|Split|C_{max}$, the mathematical model of the robust discrete min-max regret version of $Rm|Split|C_{max}$ has the same number of decision variables and a set of additional constraints related to the discrete scenarios. Hence, and based on Khachian (1979); Karmarkar (1984), we have the following result:

Corollary 4.1.2 *DmMR$(Rm|Split|C_{max}; p_{ij})$, the robust version the polynomial-time solvable problem $Rm|Split|C_{max}$, is also polynomial.*

Example 4.1.2 *We solve $LP_{r-Split}$ the robust discrete formulation of min-max regret version of $Rm|Split|C_{max}$. The optimal robust solution minimizing the maximal regret X_r is represented in Table 4.3. Under this robust solution: 57% of $J1$ is processed on machine 1, and 43% on machine 2. $J2$ is processed on machine 1 for 25%, and for 75% on machine 3. 47% of $J3$ is processed on machine 1, 21% is processed on machine 2 and 32% is processed on machine 3. And finally, 40% of $J4$ is processed on machine 2 and 60% is processed on machine 3. The optimal regret is equal to 3.5 units. Under 4 scenarios: s_1, s_2, s_3 and s_5, the regret is equal to 3.5 units while it is equal to 3.22 under s_6 and 2.82 under s_4 (See Figure 4.2).*

TABLE 4.3: An optimal robust solution minimizing the maximal regret in the splitting case: (X_r)

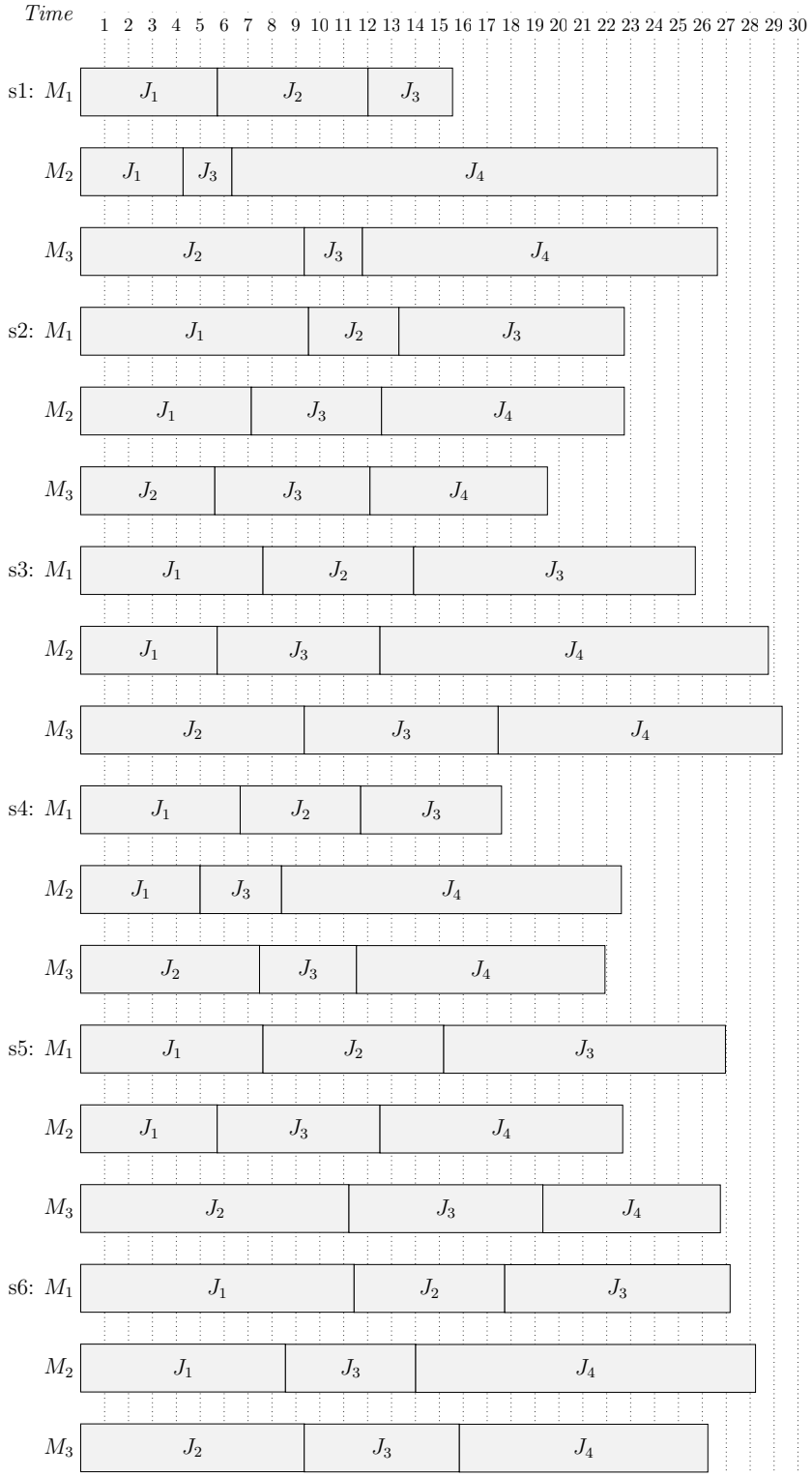
	J_1	J_2	J_3	J_4
M_1	0.57	0.25	0.47	0
M_2	0.43	0	0.21	0.40
M_3	0	0.75	0.32	0.6

TABLE 4.4: The worst-case makespan gaps under X_r in the splitting case

X	*Worst-case makespan*	Gap$(\%)$
$X_{s_{nominal}}$	45.62	55.4%
X_{max}	29.16	-0.6$\%$
$X_{average}$	29.11	-0.7$\%$

We notice in this example that the worst-case makespan of the optimal robust solution X_r is less expensive compared to the worst-case makespan of the nominal solution but its is quite similar to the worst-case makespan of the robust artificial scenario solutions X_{max} and $X_{average}$ (See Table 4.4). We will examine if this property could be generalized or not through the computational results.

FIGURE 4.2: Potential scenarios scheduled according to X_r in the splitting case



In this section, we have formulated the robust versions of the problem $Rm|Split|C_{max}$ using two different objectives and we have shown that the robust solutions in the example have led to good improvements. In the next section, we have carried out extensive numerical experiments to test the performance of the proposed robust formulations.

4.2 Computational results

In these numerical experiments, we consider the different problem sizes and the three uncertainty degrees introduced in the test protocol Chapter 3. The objectives of the computational results are summarized in 2 points:

1. We report the CPU times required to solve the robust formulations for different problem sizes.
2. We analyse the robustness cost of the optimal robust solutions (as given in Chapter 3). We also compare the optimal robust solutions to X_{max} and $X_{average}$ solutions.

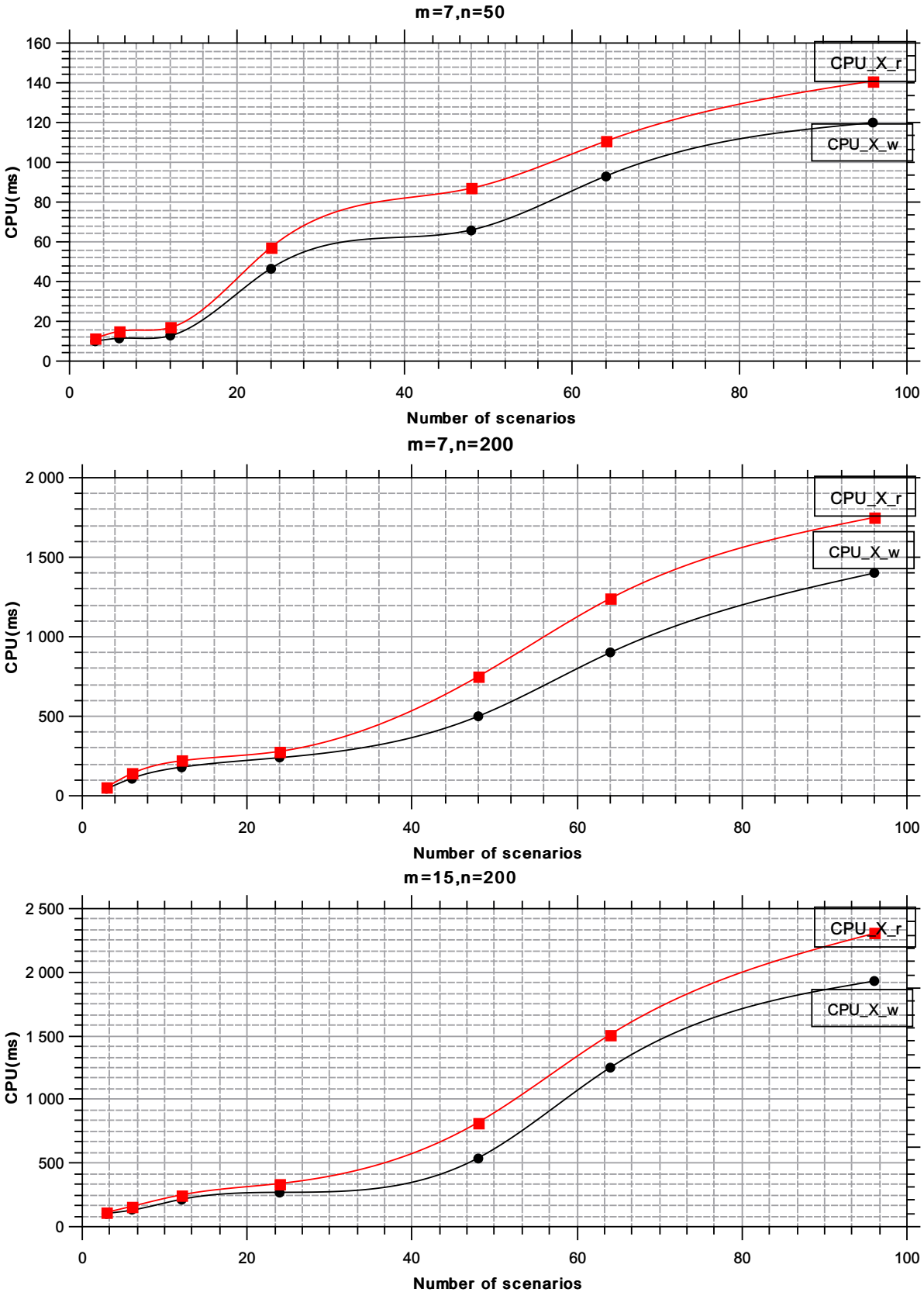
TABLE 4.5: CPU times for the optimal worst-case makespan computation under different problem sizes

		CPU(ms)						
m	n	$\|S\|=3$	$\|S\|=6$	$\|S\|=12$	$\|S\|=24$	$\|S\|=48$	$\|S\|=64$	$\|S\|=96$
7	10	2.1	3.4	3.7	5.8	7.3	9.6	37.9
	50	10.0	11.5	12.7	46.5	66.0	93.0	110.6
	200	35.2	45.7	119.3	155.9	468.0	591.2	791.2
15	15	3.1	5.4	7.5	9.2	9.3	11.6	56.2
	50	30.0	45.5	62.7	96.2	116	154	278.5
	200	105.2	130.9	215.9	270.5	540.4	1250.9	1930.5

TABLE 4.6: CPU times for the optimal regret computation under different problem sizes

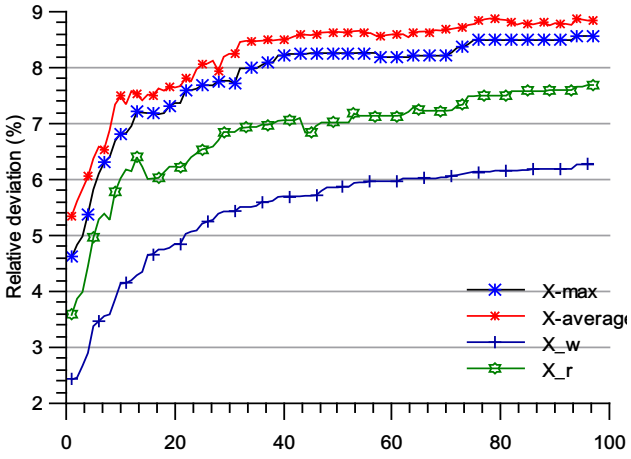
		CPU(ms)						
m	n	$\|S\|=3$	$\|S\|=6$	$\|S\|=12$	$\|S\|=24$	$\|S\|=48$	$\|S\|=64$	$\|S\|=96$
7	10	2.9	4.1	4.7	6.3	7.3	10.4	42.5
	50	12.0	15.4	16.3	57.2	87.0	112.0	147.5
	200	45.7	56.2	123.4	176.1	494.0	620.4	810.2
15	15	4.6	6.6	8.8	11.2	13.2	14.6	69.1
	50	42.0	62.5	62.7	96.2	130	173	298.2
	200	114.1	160.1	250.2	346.7	892.3	1506.5	2305.2

FIGURE 4.3: CPU times of the robust formulations resolution in the case of splitting

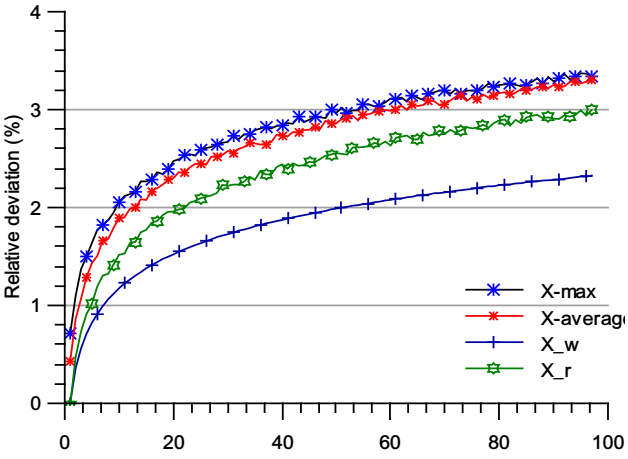


With respect to point (1), the results regarding the computational times for varying problem sizes have been summarized in Table 4.6. The reported CPU times clearly illustrate the effect of the problem size on the CPU times. We notice that the computational times increases highly as the size of the deterministic problem increases while the curve maintain the same trends. For both worst-case makespan and maximal regret, the computational time increases as the number of scenarios increases. As expected, the computational times required to obtain the optimal regret are more important than the computational time required to obtain the optimal worst-case makespan. But, we can observe that the CPU times demonstrate that the approach is able to solve medium-sized and high-sized instances with modest computational efforts.

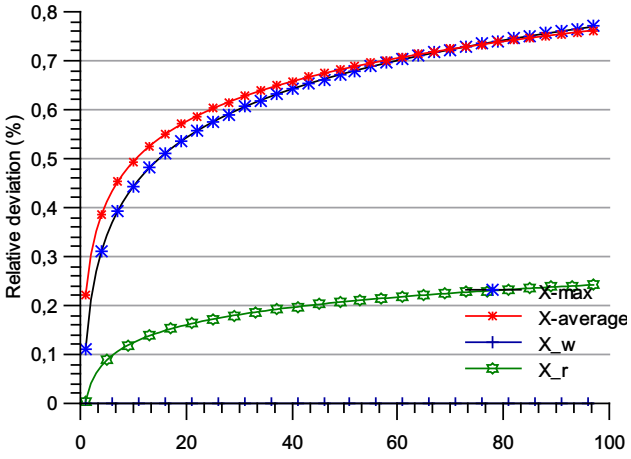
FIGURE 4.4: Optimal robust solution worst-case makespans in the splitting case: small size workshops under low uncertainty



a. (m=3,n=10,d=1)



b. (m=3,n=50,d=1)



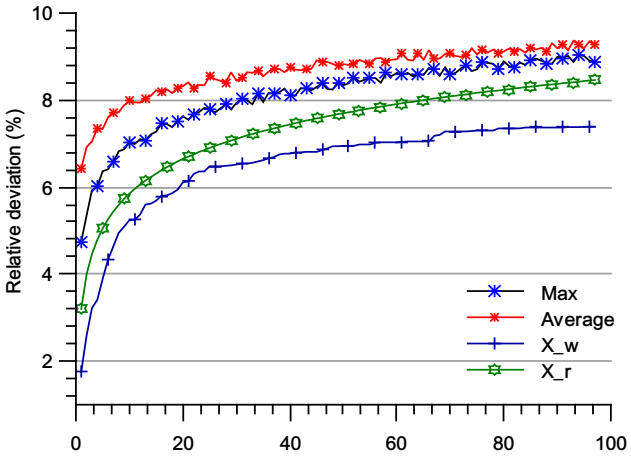
c. (m=3,n=200,d=1)

3 machines	n	y	R^2
X_{max}	10	0.99ln(x) + 3.37	0.97
	50	0.58ln(x) + 0.49	0.97
	200	0.14ln(x) + 0.11	0.97
$X_{average}$	10	0.93ln(x) + 3.76	0.95
	50	0.62ln(x) + 0.41	0.96
	200	0.14ln(x) + 0.08	0.95
X_w	10	1.06ln(x)+1.07	0.96
	50	0.50ln(x) + 0.006	0.95
	200	0.00515ln(x)	0.96
X_r	10	0.69ln(x)+2.7	0.96
	50	0.65ln(x) + 0.004	0.97
	200	0.05ln(x) + 0.005	0.96

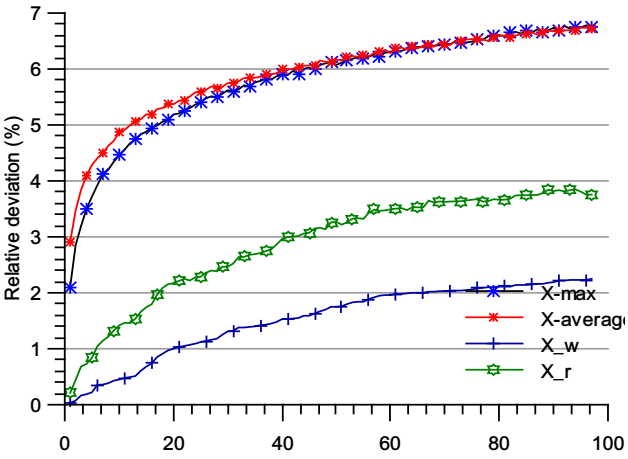
d. logarithmic trends

With respect to point (2), we address the evolution of the robustness cost of the optimal robust solutions according to the number of discrete scenarios introduced in Chapter 3. We analyse the results according to the problem size and we distinguish the three uncertainty degrees. The first observation that can be made is that the worst-case relative deviation of the optimal robust solutions minimizing the worst-case makespan X_w (*resp.* the maximal regret X_r) also follows a logarithmic line-trend. Moreover, we can easily remark that these solutions highly reduce the robustness cost in comparison with the artificial scenario solutions. For instance, ***under low uncertainty degree***, in small size workshops under a low job variety (Figure 4.4

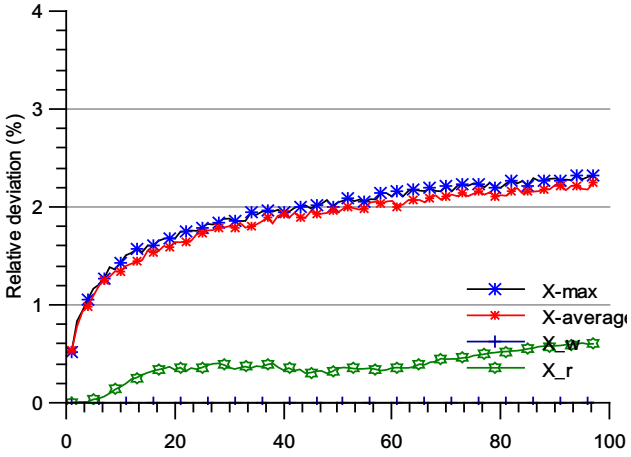
FIGURE 4.5: Optimal robust solution worst-case makespans in the splitting case: medium size workshops under low uncertainty



a. (m=7,n=10,d=1)



b. (m=7,n=50,d=1)



c. (m=7,n=200,d=1)

m=7	n	y	R^2
X_{max}	10	0.91ln(x) + 4.82	0.96
	50	0.8ln(x) + 2.09	0.96
	200	0.38ln(x) + 0.54	0.96
$X_{average}$	10	0.60ln(x) + 6.47	0.96
	50	0.53ln(x) + 2.90	0.96
	200	0.37ln(x) + 0.51	0.97
X_w	10	1.29ln(x) +2.3	0.95
	50	0.75ln(x) + 0.1	0.97
	200	0	0.96
X_r	10	1.49ln(x) + 3.8	0.96
	50	1.04ln(x) + 0.3	0.97
	200	0.32ln(x) + 0.06	0.96

d. logarithmic trends

graph a. and table d.), the worst-case relative deviation of X_w is comprised between 1% and 6%. Its initial

robustness cost is around 1% and the increase rate is also around 1%. The improvement in comparison with X_{max} is very notifiable especially concerning the initial robustness cost (1% instead of 4%) which makes the robustness cost 4 times less expensive. The robustness cost of X_r is comprised between 2.7% and 7%. Its initial robustness cost is around 2.7% and the increase rate is around 1%. Here again, the improvement in comparison with X_{max} is notifiable, the initial robustness cost is equal to 2.7% instead of 4%. Furthermore, ***we notice that the robustness cost of the robust optimal solutions become more and more interesting as the number of jobs increases***. For instance, under high job variety (Figure 4.4 graph c. and table d.), the robustness cost of X_w is almost null while the initial robustness cost and the increase rate of X_r are negligible: the robustness cost of X_r is comprised between 0.01% and 0.25%.

When we increase the number of machines, the robustness costs of the robust optimal solutions increase as it was the case for the artificial scenario solutions. For example, in medium size workshops under low job variety (Figure 4.5 graph a. and table d.), the robustness cost of X_w is comprised between 2.3% and 7%. Its initial robustness cost is around 2.3% and its increase rate is around 1.29%. The improvement in comparison with X_{max} is still significant as the initial robustness cost is equal to 2.3% under X_w instead of 4.82% under X_{max} but, we can observe that the ratio has decreased: the robustness cost is only 2 times inferior. The of X_r is comprised between 3.8% and 8.2%. Its initial robustness cost is around 3.8% and the increase rate is around 1.5%. Once again, there is an improvement when we apply X_r instead of X_{max}, but the improvement ratio has decreased (only 1.26 times less expensive).

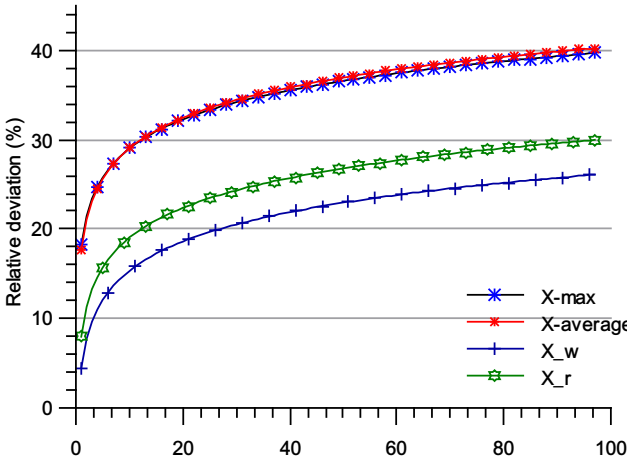
Under medium and high degrees of uncertainty, the the robustness costs of X_w and X_r increase but their trend lines are also logarithmic. For instance, under medium uncertainty in small size workshops under low job variety (Figure 4.6 graph a. and table d.), the worst-case relative deviation of X_w is comprised between 4.44% and 27%. Its initial robustness cost is around 4.4% and the increase rate is also around 4.73%. In this case, the improvement in comparison with X_{max} is very significant, we observe that the initial robustness cost of X_w is more than four times less expensive (4.4% instead of 18.22%). The robustness cost of X_r is comprised between 7.9% and 30%. Its initial robustness cost is around 8% and the increase rate is around 4.8%.

Furthermore, ***under high uncertainty degree***, in small size workshops under low job variety (Figure E.4 graph a. and table d.), the initial robustness cost of X_w is around 9%. The rate change is around 11%. The initial robustness cost of X_r is around 13% and its increase rate is around 3.4%. In both cases, the improvement in comparison with artificial scenario solutions is very significant. The initial robustness cost is 5 times less expensive when we use X_w instead of X_{max} and it is 3 times less expensive when we use X_r instead of X_{max}.

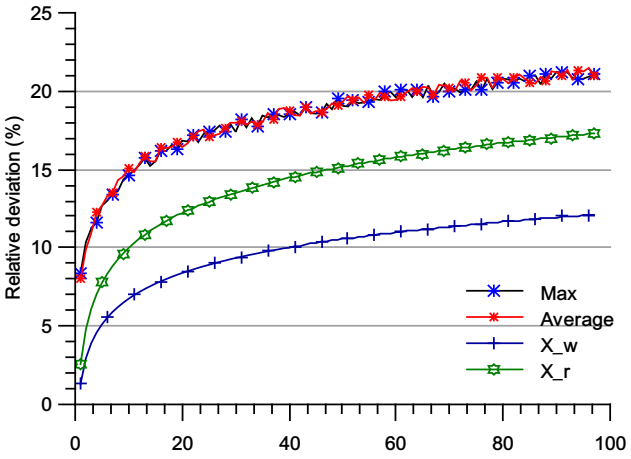
The curves in high size workshops lead to the same conclusions (See Appendix E).

Overall, the robustness costs of the robust optimal solution are very interesting. The initial robustness costs are relatively low besides the increase rates are insignificant. These properties are more accurate when the variety of jobs is important while the number of machines is small. In these special cases, the robust optimal solutions are very favourable and not expensive compared to the nominal solution under the deterministic case.

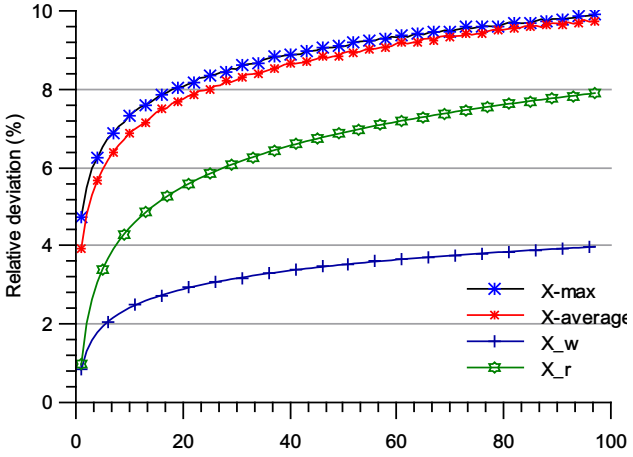
FIGURE 4.6: Optimal robust solution worst-case makespans in the splitting case: small size workshops under medium uncertainty



a. (m=3,n=10,d=2)



b. (m=3,n=50,d=2)



c. (m=3,n=200,d=2)

3 machines	n	y	R^2
X_{max}	10	4.69ln(x) + 18.22	0.98
	50	2.85ln(x) + 7.98	0.95
	200	1.12ln(x) + 4.71	0.978
$X_{average}$	10	4.93ln(x) + 17.68	0.95
	50	2.85ln(x) + 8.03	0.96
	200	1.27ln(x) + 3.90	0.97
X_w	10	4.73ln(x)+4.44	0.96
	50	2.36ln(x)+1.31	0.95
	200	0.68ln(x)+0.83	0.97
X_r	10	4.8ln(x)+7.96	0.96
	50	3.22ln(x)+2.56	0.97
	200	1.51ln(x)+0.96	0.96

d. logarithmic trends

4.3 Robust makespan on uniform and identical parallel machine scheduling problems with splitting under discrete processing time scenarios

In this section, we are interested in the robust makespan on uniform parallel machines Q and identical parallel machines P with splitting. Given that Q and P are special cases of R and based on the complexity results of *DmM*$(Rm|Split|C_{max};p_{ij})$ and *DmMR* $(Rm|Split|C_{max};p_{ij})$, we deduce that the robust versions *DmM*$(\alpha|Split|C_{max};p_{ij})$ and *DmMR*$(\alpha|Split|C_{max};p_{ij})$ are polynomial when $\alpha \in \{Qm, Pm\}$. Furthermore, we will show in this section that their optimal robust solutions can be identified without solving the

linear programs when the uncertainty of processing times is due to the uncertainty of demands.

4.3.1 Discrete min max versions of $Qm|Split|C_{max}$ under discrete processing time scenarios

When the parallel machines are uniform, the machine speeds are not job dependent:

$$p_{ij} = p_j/v_i, \forall i \in M, \forall j \in N.$$

where v_i is the speed of machine i and p_j the processing requirement of job j.

In the deterministic case, there exists an optimal solution that consists in splitting each job into m subjobs, the size of the subjob of job j to assign to machine i is equal to the ratio $\frac{v_i}{\sum_{i=1}^{m} v_i}$. The optimal assignment solution X^* of $Qm|Split|C_{max}$ depends only on machine speeds.

$$x^*_{ij} \quad = \frac{v_i}{\sum_{i=1}^{m} v_i}, \forall i \in M, \forall j \in N. \tag{4.3}$$

Consequently, the optimal makespan is such that:

$$\begin{aligned} C^*_{max}(X^*) &= \sum_{j=1}^{n} p_{ij} x^*_{ij} \\ &= \sum_{j=1}^{n} (\frac{p_j}{v_i}) \frac{v_i}{\sum_{i=1}^{m} v_i} \\ &= \frac{\sum_{j=1}^{n} p_j}{\sum_{i=1}^{m} v_i} \end{aligned}$$

The deterministic makespan calculation is done in $O(n)$.

Case1: If the processing times are uncertain due to the uncertainty of job requirements, we will denote:

$$p^s_{ij} = (p^s_j)/v_i$$

where p^s_j is the processing requirement of job j under scenario s.

Given that the optimal solution X^* depends only on machine speeds (See equation 4.3), X^* is also an optimal solution for any scenario from S. The worst-case makespan of X^* is then:

$$\begin{aligned} \max_{s_k \in S} C^{s_k}_{max}(X^*) &= \max_{s_k \in S} \frac{\sum_{j=1}^{n} p_j^{s_k}}{\sum_{i=1}^{m} v_i} \\ &= \frac{1}{\sum_{i=1}^{m} v_i} \max_{s_k \in S} \sum_{j=1}^{n} p_j^{s_k}. \end{aligned}$$

Based on this equality, the worst-case scenario s^w is the scenario where the total quantity of demand over jobs is maximal. Consequently, we have the following results:

Corollary 4.3.1 *DmM$(Qm|Split|C_{max};p_j)$ could be solved to optimality in $O(n \mid S \mid)$ time by mean of the optimal deterministic solution X^* where: $x^*_{ij} = \frac{v_i}{\sum_{i=1}^{m} v_i}$ $\forall i \in M$, $\forall j \in N$.*

The worst-case makespan is equal to the makespan of X^* applied to the worst-case scenario s_w where the total of job processing requirements is maximal..

Moreover, the maximal deviation of the solution X^* over scenarios is

$$\begin{aligned}\max_{s_k \in S}\{C^{s_k}_{max}(X^*) - C^{s_k*}_{max}\} &= \max_{s_k \in S}\{C^{s_k*}_{max} - C^{s_k*}_{max}\} \\ &= 0\end{aligned}$$

which gives the following result:

Corollary 4.3.2 *DmMR$(Qm|Split|C_{max};p_j)$ could be solved to optimality in constant time by mean of the deterministic optimal solution X^* that depends only on machine speed such that $x^*_{ij} = \frac{v_i}{\sum_{i=1}^{m} v_i}$, and the maximal deviation of X^* is equal to zero.*

4.3.2 Discrete min max versions of $Pm|Split|C_{max}$ under discrete processing time scenarios

When the parallel machines are identical $p_{ij} = p_j$. We can then draw the same results as for the case uniform parallel machines with further simplification.

In the deterministic case, there exists an optimal assignment solution that consists in splitting each job j into m subjobs, the size of the subjob of j to assign to each machine i is equal to $\frac{p_j}{m}$.

$$x^*_{ij} = \frac{1}{m}, \forall i \in M, \forall j \in N.$$

Consequently:

$$\begin{aligned}C^*_{max}(X^*) &= \sum_{j=1}^{n} p_j x^*_{ij} \\ &= \sum_{j=1}^{n} p_j \frac{1}{m} \\ &= \frac{1}{m}\sum_{j=1}^{n} p_j.\end{aligned}$$

The deterministic makespan calculation is done in $O(n)$.

If the processing times are uncertain and the uncertainty is due to job requirements, we will denote:

$$p^s_{ij} = p^s_j$$

Given that X^* depends only on machines number, X^* is also an optimal solution for any scenario from S. The worst-case makespan of X^* is then:

$$\begin{aligned}\max_{s_k \in S} C_{max}^{s_k}(X) &= \max_{s_k \in S} \sum_{j=1}^{n} \frac{1}{m} p_j^{s_k} \\ &= \frac{1}{m} \max_{s_k \in S} \sum_{j=1}^{n} p_j^{s_k}\end{aligned}$$

Based on this equality, the worst-case scenario s_w is the scenario where the total quantity of demand over jobs is maximal. Consequently, we have the following result:

Corollary 4.3.3 *DmM$(Pm|Split|C_{max}; p_j)$ could be solved to optimality in $O(n \mid S \mid)$ time by mean of the optimal deterministic solution X^* where all the jobs are split into m equal sub-jobs:*

x_{ij}^*= $\frac{1}{m}$, $\forall i \in M, \forall j \in N$.

And the worst-case makespan is equal to the makespan of X^* applied to the worst-case scenario s_w where the total of job processing requirements is maximal..

The maximal regret of the solution X^* over scenarios is

$$\begin{aligned}\max_{s_k \in S}\{C_{max}^{s_k}(X^*) - C_{max}^{s_k *}\} &= \max_{s_k \in S}\{\frac{\sum_{j=1}^{n} p_j^{s_k}}{m} - \frac{\sum_{j=1}^{n} p_j^{s_k}}{m}\} \\ &= 0\end{aligned}$$

Thus, we have the following results:

Corollary 4.3.4 *DmMR$(Pm|Split|C_{max}; p_j)$ could be solved to optimality in constant time by mean of the deterministic optimal solution X^* where all the jobs are split into m equal sub-jobs, and the maximal deviation of X^* is equal to zero.*

Conclusion

In this chapter, we have applied the robust discrete optimization framework developed by Kouvelis and Yu (1997) to compute the optimal robust solutions for the makespan minimization on unrelated parallel machines with job splitting under uncertain processing times. We showed that the direct application of robust optimization techniques led to no solution. To overcome this difficulty, we proposed a formulation in which the uncertainty appears in the objective instead of the assignment constraints. For this purpose, We used assignment ratios as decision variables instead of temporal variables, to insure the feasibility. We provide robust solutions in polynomial time using respectively the minimization of the worst-case makespan and respectively the minimization of the maximal regret. We also considered the special cases of uniform and identical parallel machines. The complexity results in the case of splitting are summarized in the table below:

TABLE 4.7: Complexity results in the case of splitting

DmM(R)$(\alpha\|\beta\|\gamma;\theta)$	*Complexities*	*References*
DmM$(Rm\|Split\|C_{max};p_{ij})$	Polynomial	Section 4.1.1
DmMR$(Rm\|Split\|C_{max};p_{ij})$	Polynomial	Section 4.1.2
DmM$(Qm\|Split\|C_{max};p_{ij})$	Polynomial	Section 4.3.1
DmMR$(Qm\|Split\|C_{max};p_{ij})$	Polynomial	Section 4.3.1
DmM$(Pm\|Split\|C_{max};p_{ij})$	Polynomial	Section 4.3.2
DmMR$(Pm\|Split\|C_{max};p_{ij})$	Polynomial	Section 4.3.2

For the unrelated parallel machines, the computational results show that the robust formulations can be solved with modest CPU times even for large size instances. The CPU time required to minimize the maximal regret is not so much higher than the CPU time required to minimize the worst-case makespan. Besides, the computational results show that the robustness costs of the optimal robust solutions minimizing the worst-case makespan (*resp.* the maximal regret) also follow logarithmic trend lines. They increase when we increase the number of machines, the number of scenarios or the degree of uncertainty. But, they decrease when we increase the number of jobs. Moreover, these solutions reduce considerably the robustness cost compared to X_{max} and $X_{average}$ solutions. The optimal robust solution that minimizes the worst-case makespan X_w leads to very interesting results in term of robustness cost, followed by the optimal robust solution that minimizes the maximal regret X_r.

In the next chapter, we extend the robust formulations to compute optimal robust solutions when we consider the preemptive case $(Rm|pmtn|C_{max})$ under discrete processing time scenarios.

Chapter 5

Robust parallel machine scheduling with job preemption under uncertain processing times

Abstract: The goal of this chapter is to extend the robust formulations developed in Chapter 4 to compute the optimal robust solutions for the makespan minimization on unrelated parallel machines with job preemption under uncertain processing times. The major difference between the preemptive case and the splitting one is due to the computation of a sequence decision. We consider in this work that the decision-maker is interested in computing a robust assignment solution which is unique and accepts to have different sequences (one per scenario). Thus, we propose robust formulations of $(Rm|pmtn|C_{max})$ that allow to construct robust schedules based on robust assignment solutions. We respectively use the worst-case makespan minimization and the maximal regret minimization to compute robust assignment solutions. We can use preemptive open shop algorithm given by Gonzalez and Sahni (1976) to construct each scenario sequence. In the computational results, once again, we quantify the robustness costs of the optimal robust solutions and to compare them to those of the artificial scenario solutions in the case of preemption.

Contents

Introduction

In this chapter, we extend the application of the robust discrete optimization framework to the preemptive unrelated parallel machines under discrete processing time scenarios. The major difference between the preemptive case and the splitting case is due to the computation of a sequence decision. Indeed, as it was mentioned earlier in chapters 3 and 4, the computation of a robust preemptive schedule on parallel machines is composed of two steps: the computation of a duration matrix and then the construction of a feasible sequence.

Under a set of discrete potential scenarios, some decision-makers might be interested in computing a robust assignment solution which is unique. In which concerns the sequence, they may tolerate to have different sequences (one per scenario). In Section 5.1, we propose robust formulations of $(Rm|pmtn|C_{max})$ that allow to answer the former concern. In Section 5.2, we present the computational results that allow to quantify the robustness cost of the optimal robust solutions and to compare them to the artificial scenario solutions.

5.1 Robust schedules based on the robust assignment solutions: A sequential approach

When decision makers are interested in computing a unique assignment solution and they accept to have different sequences and permutations (one per scenario), the problem can be solved in two steps as follows:

- The first step consists in computing a robust assignment solution x^*_{ij} in a proactive way as done in the case of splitting. This solution can be obtained by solving different robust formulations (see Sections 5.1.1 and 5.1.2). Therefore, we propose to change the decision variables in the static preemptive problem proposed by Lawler and Labetoulle (1978). We consider x_{ij} as the assignment ratio of job j to machine i: x_{ij}=$\frac{t_{ij}}{p_{ij}}$. This transformation will lead to the deterministic linear programming ($LP_{pmtn-assign}$) below:

Minimize C_{max}

Subject to

$$\sum_{i=1}^{m} x_{ij} = 1, \qquad j = 1, \ldots, n,\ s = s_1, \ldots, s_k$$

$$\sum_{j=1}^{n} p_{ij} x_{ij} - C_{max} \leq 0, \qquad i = 1, \ldots, m$$

$$\sum_{i=1}^{n} p_{ij} x_{ij} - C_{max} \leq 0, \qquad j = 1, \ldots, n$$

$$0 \leq x_{ij} \leq 1, \qquad i = 1, \ldots, m\ j = 1, \ldots, n$$

$LP_{pmtn-assign}$ will return assignment ratios as decision variables instead of temporal durations. Based on this, we will formulate and solve the robust versions of the preemptive problem by extending the robust formulations given in the case of splitting (see Chapter4). The proofs are very similar.

- The second step consists in constructing a sequence for each scenario. This step can be postponed until the real scenario is revealed, or it can be executed in a proactive way under a reasonable number scenarios. We compute the duration matrix of each scenario according to the robust assignment solution as: $d_{ij}^s = p_{ij}^s x_{ij}^*$ and then, we solve for each scenario s a preemptive open shop problem ($Om|pmtn|C_{max}$) according to (Gonzalez and Sahni (1976))'s algorithm by considering the durations d_{ij}^ss as data entries.

In Section 5.1.1, we develop the robust assignment formulation to compute a solution minimizing the worst-case makespan. And in Section 5.1.2, we develop robust assignment formulation to compute a solution minimizing the maximal regret.

5.1.1 Formulation of the min-max version of $Rm|pmtn|C_{max}$ under discrete processing time scenarios

To provide a robust assignment solution minimizing the worst-case makespan, we can solve the discrete min-max version of $Rm|pmtn|C_{max}$ that we formulate as follows (LP_{w-pmtn}):

$$\text{Minimize } C_w$$

Subject to

$$\sum_{i=1}^{m} x_{ij} = 1, \qquad j = 1, \ldots, n \tag{5.1}$$

$$\sum_{j=1}^{n} p_{ij}^s x_{ij} - C_w \leq 0, \qquad i = 1, \ldots, m,\ s = s_1, \ldots, s_k \tag{5.2}$$

$$\sum_{i=1}^{m} p_{ij}^s x_{ij} - C_w \leq 0, \qquad j = 1, \ldots, n,\ s = s_1, \ldots, s_k \tag{5.3}$$

$$0 \leq x_{ij} \leq 1, \qquad i = 1, \ldots, m,\ j = 1, \ldots, n \tag{5.4}$$

In this formulation, x_{ij} is the robust assignment ratio of job j to machine i that minimizes the worst-case makespan C_w over all the scenarios (the proof is similar to the proof of the $Rm|Split|Cmax$).

The mathematical formulation of the robust discrete min max version of $Rm|pmtn|C_{max}$ for the computation of the robust assignment solution is an LP. Thus, the assignment problem is polynomial based on Khachian (1979); Karmarkar (1984). Moreover, for each scenario s the construction of the resolution of the preemptive open shop is polynomial, solvable in $O(r + \min(r^2, n^4, m^4))$ where r is the number of nonzero elements in X the matrix of assignments x_{ij}'s, and consequently:

Corollary 5.1.1 *DmM($Rm|pmtn|C_{max}; p_{ij}$), the robust version of the polynomial-time solvable $Rm|pmtn|C_{max}$, is polynomial.*

Example 5.1.1 *By solving LP_{w-pmtn}, the optimal robust assignment solution over the six scenarios (cf. Figure 5.1) has a worst-case makespan of 28.92 units. Under this solution, $J1$ is processed for 43% on machine 1 and for 57% on machine 2. $J2$ is processed for 52% on machine 1, and for 48% on machine 3.*

$J3$ is processed for 31% on machine 1 and for 45% on machine 2 and for 24% on machine 3 while J4 is processed for 16% on machine 2 and for 84% on machine 3 (see Table 5.1).

TABLE 5.1: An optimal solution minimizing the worst-case makespan in the preemptive case: (X_w)

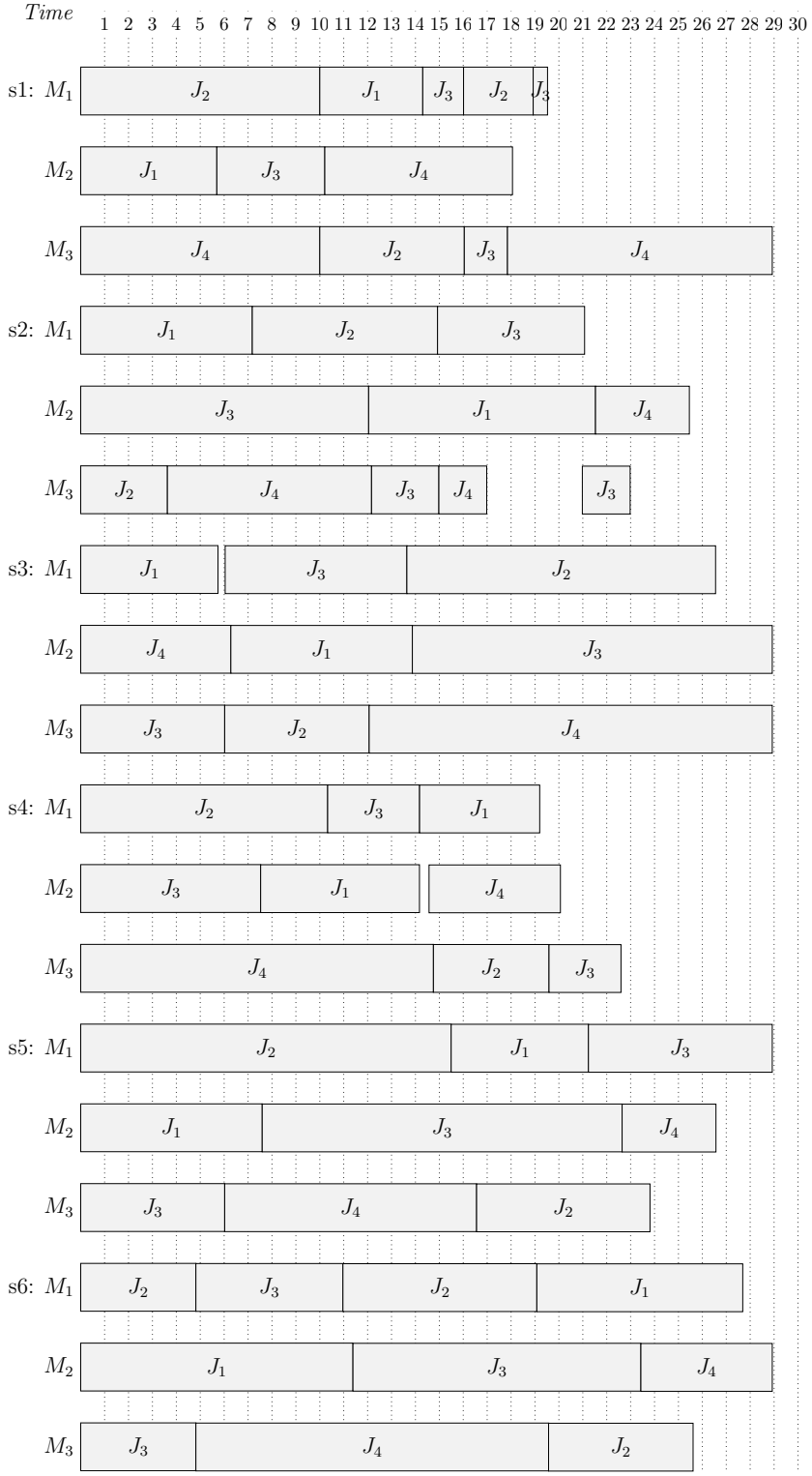
X_w^*	J_1	J_2	J_3	J_4
M_1	0.43	0.52	0.31	0
M_2	0.57	0	0.45	0.16
M_3	0	0.48	0.24	0.84

TABLE 5.2: The worst-case makespan gaps under X_w in the preemptive case

X	*Worst-case makespan*	Gap($\%$)
$X_{s_{nominal}}$	46.67	61%
X_{max}	31.25	7.8%
$X_{average}$	32.5	12.1%

Based on the robust assignment ratio solution X_w, we construct the duration matrices corresponding to the six potential scenarios. Each scenario can be scheduled independently according to the robust assignment solution without exceeding the robust makespan. For this purpose, we can solve the six independent preemptive open shop problems. These schedules are represented in Figure 5.1. Compared to the nominal scenario solution $X_{s_{nominal}}$ and the artificial scenario solutions X_{max} and $X_{average}$, the worst-case makespan has been reduced under X_w (see Table 5.2).

FIGURE 5.1: Potential scenarios scheduled according to X_w in the preemptive case



5.1.2 Formulation of the min-max regret version of $Rm|pmtn|C_{max}$ under discrete processing time scenarios

We can also provide a robust assignment solution by solving the min-max regret version of $Rm|pmtn|C_{max}$ under discrete processing time scenarios. Based on the proof given in Section (4.1.2) and by considering the constraints that prohibit the overlapping, we can formulate the problem as (LP_{r-pmtn}):

$$\text{Minimize } r_{max}$$

Subject to

$$\begin{aligned}
\sum_{i=1}^{m} x_{ij} &= 1, && j = 1, \ldots, n \\
\sum_{j=1}^{n} {p_{ij}}^s x_{ij} - C_{max}^{s*} &\leq r_{max}, && i = 1, \ldots, m,\ s = s_1, \ldots, s_k \\
\sum_{i=1}^{n} {p_{ij}}^s x_{ij} - C_{max}^{s*} &\leq r_{max}, && j = 1, \ldots, n,\ s = s_1, \ldots, s_k \\
0 \leq x_{ij} &\leq 1, && i = 1, \ldots, m,\ j = 1, \ldots, n.
\end{aligned}$$

In this formulation, x_{ij} is the robust assignment ratio of job j to machine i that minimizes the maximal regret r_{max} over all the scenarios. C_{max}^{s*} is the optimal makespan under scenario s. For each scenario s, we solve the deterministic problem $Rm|pmtn|C_{max}^{s}$ under the scenario s to obtain C_{max}^{s*}.

The computation of the robust assignment solution minimizing the maximal regret for the preemptive unrelated parallel machine is an LP. The optimal deterministic solutions of the discrete scenarios are already computed by solving the deterministic LPs of each scenario. Thus, the assignment problem is polynomial based on Khachian (1979); Karmarkar (1984). Moreover, for each scenario s the construction of the resolution of the preemptive open shop is polynomial, solvable in $O(r + \min(r^2, n^4, m^4)$ where r is the number of nonzero elements in X the matrix of assignments x_{ij}'s, which leads to the following:

Corollary 5.1.2 *DmMR*$(Rm|pmtn|C_{max}; p_{ij})$, *the robust version the polynomial-time solvable* $Rm|pmtn|C_{max}$, *is also polynomial.*

Example 5.1.2 *By solving* LP_{r-pmtn}, *the optimal robust solution* X_r *that minimizes the maximal regret has a maximal regret equal to* 4.02 *units. The assignment ratios according to this solution are as follows: J1 is processed on machine 1 and 2 for* 58% *and* 42% *respectively. J2 is processed on machine 1 for* 43%, *on machine 2 for* 1% *and on machine 3 for* 47%. *J3 is processed for* 34% *on machine 1, for* 37% *on machine 2 and for* 29% *on machine 3. And finally, J4 is processed for* 16% *on machine 2 and for* 84% *on machine 3. Each scenario can be scheduled independently according to the robust assignment solution* X_r.

For this purpose, we construct the duration matrices corresponding to the six potential scenarios and solve their corresponding preemptive open shop problems. These schedules are represented in Figure 5.2:

FIGURE 5.2: Potential scenarios scheduled according to X_r in the preemptive case

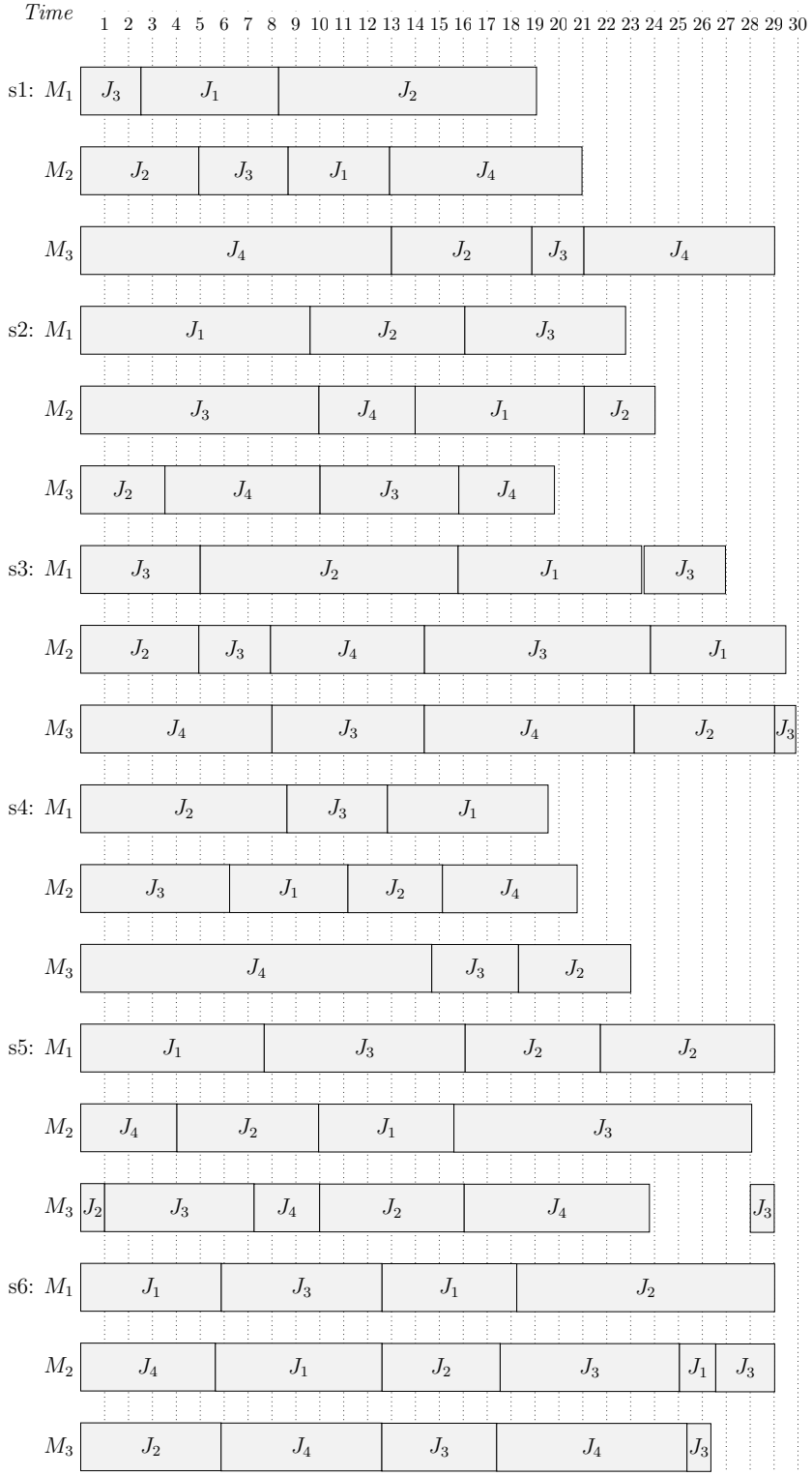


TABLE 5.3: An optimal solution minimizing the maximal regret in the preemptive case: (X_r)

X_r	J_1	J_2	J_3	J_4
M_1	0.58	0.43	0.34	0
M_2	0.42	0.1	0.37	0.16
M_3	0	0.47	0.29	0.84

TABLE 5.4: The worst-case makespan gaps under X_r in the preemptive case

X	*Worst-case makespan*	Gap(%)
$X_{s_{nominal}}$	46.67	55%
X_{max}	31.25	4.16%
$X_{average}$	32.5	8.3%

Compared to the nominal scenario solution $X_{s_{nominal}}$ and the artificial scenario solutions X_{max} and $X_{average}$, the worst-case makespan has also been reduced under X_r and it does not deviate too much from the worst-case under X_w (see Table 5.4).

In the next section, we report the computational results under which we focus on the robustness cost of the optimal robust solutions in the preemptive case.

5.2 Computational results

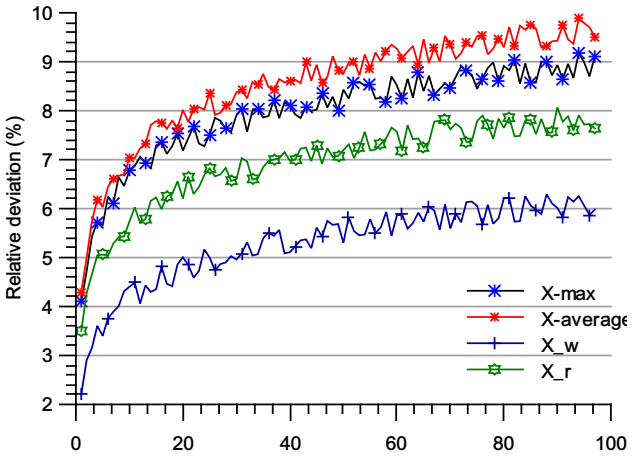
The computational tests considered in this section are the extension of the computational results presented in Chapter 3 and Chapter 4. We base on the results provided in Chapter 3 (*i.e.* the robustness costs of X_{max} and $X_{average}$) to evaluate the improvements with respect to the robust optimal solutions.

We again distinguish the three degrees of uncertainty.

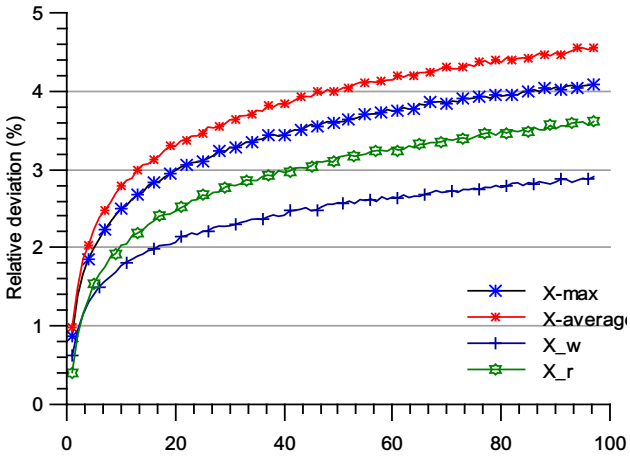
The first observations show that the robustness cost of the optimal robust solutions minimizing the worst-case makespan (*resp.* the maximal regret) also follows a logarithmic line-trend as it was the case under splitting under the three uncertainty degrees.

Under low uncertainty degree, we notice that the logarithms coefficients values are very similar to the case of splitting. For instance, in small size workshops under small job variety (Figure 5.3 graphic a. table d.), the robustness cost of X_w is comprised between 2.1% and 6.1%. Its initial robustness cost is equal to 2.2% and its increase rate is equal to also 0.85%. The improvement in comparison with X_{max} is significant as the initial robustness cost is two times less expensive. The robustness cost of X_r is comprised between 2.9% and 7%. Its initial robustness cost is around 3.5% and the increase rate is around 1.1%. The improvement under X_r in comparison with X_{max} is less important, the ratio is equal to only 1.1. ***Besides, the effect of the number of jobs increase is also less important than in the case of splitting.*** As we can observe, under high job variety (Figure 5.3 graphic c. table d.), the robustness cost of X_w is no longer null as it was the case under splitting but the values are still insignificant. Its initial cost is equal to 0.06% and its increase rate is equal to 0.05%. The improvement in comparison with X_{max} is significant as the initial robustness cost is four times less expensive. In the case of X_r, this claim is more notifiable as the initial

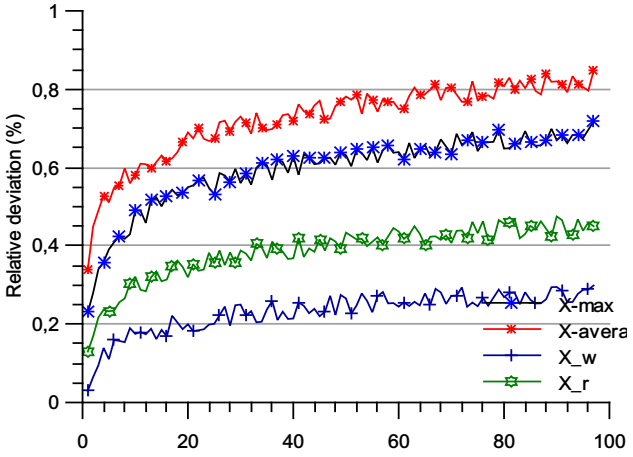
FIGURE 5.3: Optimal robust solution worst-case makespans in the splitting case: small size workshops under low uncertainty



a. (m=3,n=10,d=1)



b. (m=3,n=50,d=1)



c. (m=3,n=200,d=1)

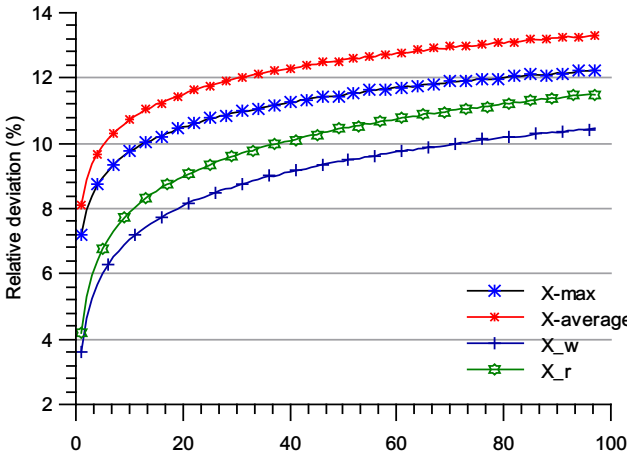
m=3	n	y	R^2
X_{max}	10	1.02ln(x)+4.2	0.97
	50	0.69ln(x) + 0.88	0.97
	200	0.16ln(x) + 0.24	0.97
$X_{average}$	10	1.16ln(x) + 4.35	0.954
	50	0.77ln(x) + 0.98	0.96
	200	0.18ln(x) + 0.37	0.98
X_w	10	0.85ln(x) + 2.22	0.961
	50	0.5ln(x) + 0.60	0.96
	200	0.05ln(x)+0.06	0.969
X_r	10	1.1ln(x)+3.5	0.98
	50	0.4ln(x) + 0.70	0.95
	200	0.07ln(x)+0.14	0.96

d. logarithmic trends

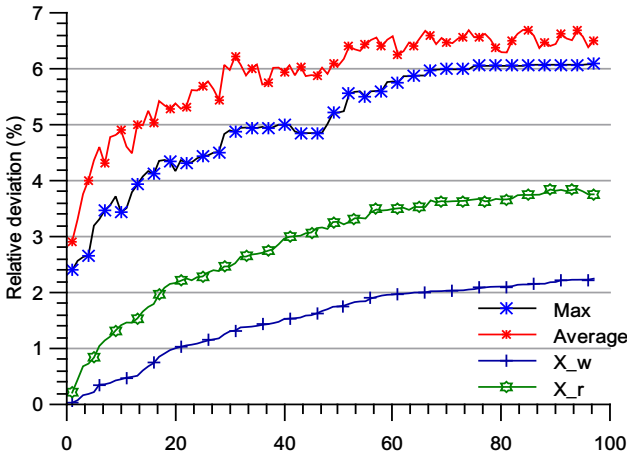
robustness cost of X_r is equal to 0.14% and the increase rate is equal to 0.07%. The improvement under X_r in comparison with X_{max} has become more important, the ratio is equal to 1.74 instead of 1.1.

When we increase the number of machines, the optimal solution robustness costs increase and this effect is more important than in the case of splitting. For instance, in medium size workshops under small job variety (Figure 5.4 graphic a. table d.), the robustness cost of X_w is comprised between 3.6% and 10.5%. Its initial robustness cost is equal to 3.6% and the increase rate is around 1.11%. The robustness cost of X_r is comprised between 4.2% and 11.8%. Its initial robustness cost 4.2% and its increase rate is around 1.6%. The improvement in comparison with X_{max} notifiable especially for X_w, the improvement ratio is equal to

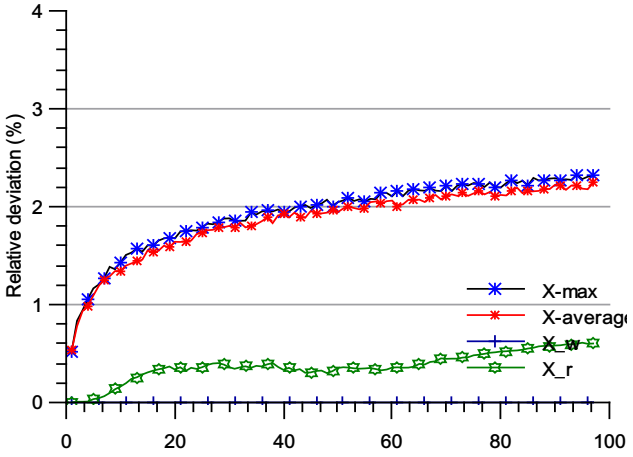
FIGURE 5.4: Optimal robust solution worst-case makespans in the preemptive case: medium size workshops under low uncertainty



a. (m=7,n=10,d=1)



b. (m=7,n=50,d=1)



c. (m=7,n=200,d=1)

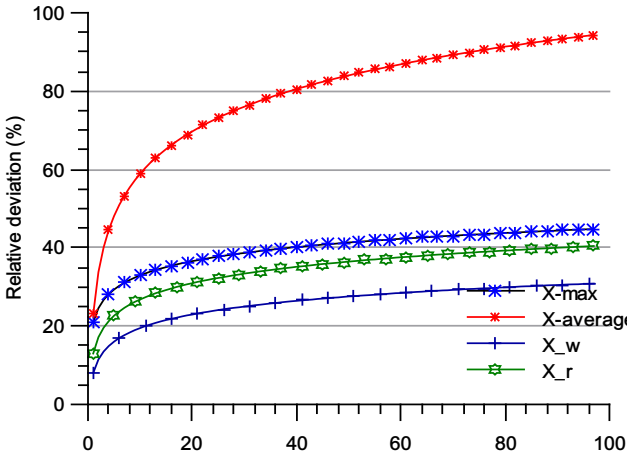
m=7	n	y	R^2
X_{max}	10	1.11ln(x) + 7.2	0.967
	50	0.92ln(x) + 3.14	0.96
	200	0.51ln(x) + 0.81	0.96
$X_{average}$	10	1.14ln(x) + 8.14	0.96
	50	0.83ln(x) + 4.59	0.96
	200	0.53ln(x) + 1.25	0.97
X_w	10	1.5ln(x) +3.6	0.95
	50	0.61ln(x) + 0.5	0.975
	200	0.06ln(x)+0.01	0.96
X_r	10	1.6ln(x) + 4.2	0.962
	50	0.9ln(x) + 2.8	0.979
	200	0.5ln(x) + 0.4	0.96

d. logarithmic trends

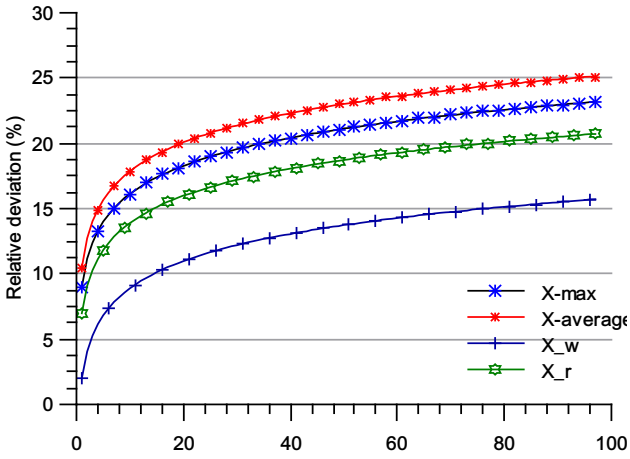
2. In the case of X_r, this ratio is equal to 1.7

When we increase the degree of uncertainty, we observe a notifiable increase in the robustness cost of the optimal solution, and the effect of uncertainty degree is very significant. For instance, ***under medium uncertainty degree***, in small size workshops under small job variety (Figure 5.5 graphic a. table d.), the initial robustness cost is equal 8.1% and its increase rate is equal to 5.3%. Here, we can observe that the improvement ratio between X_w and X_{max} become more interesting around 2.6. The robustness cost of X_r is comprised between 16.07% and 40%. Its initial robustness cost is equal to 16.07% and its increase rate is equal to 6.14%. We notice that the improvement ratio between X_r and X_{max} is around 1.3. And ***under high***

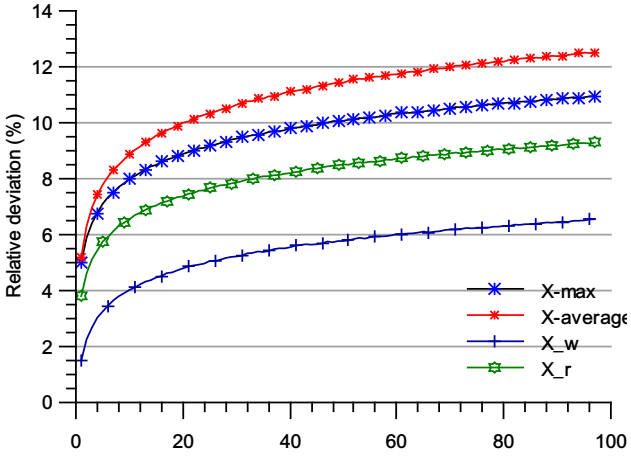
FIGURE 5.5: Optimal robust solution worst-case makespans in the preemptive case: small size workshops under medium uncertainty



a. (m=3,n=10,d=2)



b. (m=3,n=50,d=2)



c. (m=3,n=200,d=2)

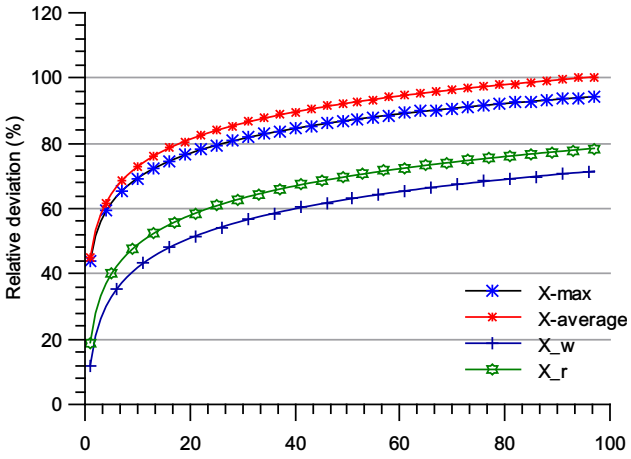
m=3	n	y	R^2
X_{max}	10	5.21ln(x) + 21.18	0.98
	50	3.26ln(x) + 9.12	0.96
	200	1.61ln(x) + 5.45	0.97
$X_{average}$	10	5.6ln(x) + 23	0.95
	50	3.24ln(x) + 10.53	0.96
	200	1.63ln(x) + 5.61	0.97
X_w	10	5.3ln(x)+8.1	0.96
	50	2.96ln(x)+2.31	0.96
	200	1.1ln(x)+1.5	0.97
X_r	10	6.14ln(x)+16.07	0.96
	50	3.27ln(x)+7.09	0.97
	200	1.2ln(x)+3.8	0.95

d. logarithmic trends

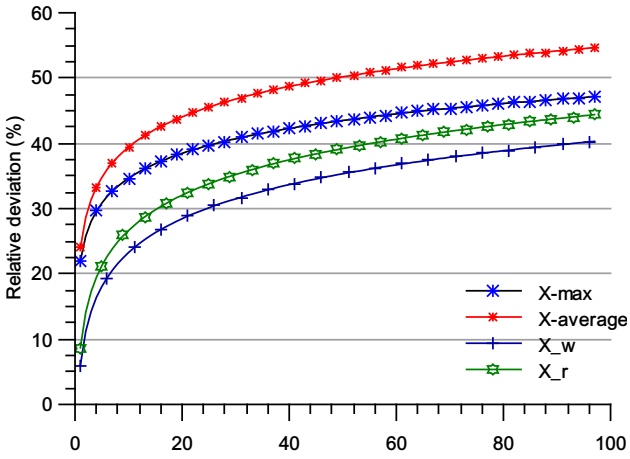
uncertainty degree, for the same problem size (Figure 5.6 graphic a. table d.), the initial robustness cost of X_w is equal to 12%. The increase rate is equal to 13%. The improvement ratio has increased under X_w and it is around 3.6 which means that X_w is 3.6 times less expensive than X_{max}. For X_r, the initial robustness cost is equal to 15% and its increase rate is equal to 13.1%. We notice here that X_r is 3 times less expensive than X_{max}.

Globally, when the variety of jobs is important while the number of machines is small, the robust optimal solutions are very favourable and not expensive compared to the nominal solution in the deterministic case. This observation is due to the fact that when the number of jobs is important while the number of machines is

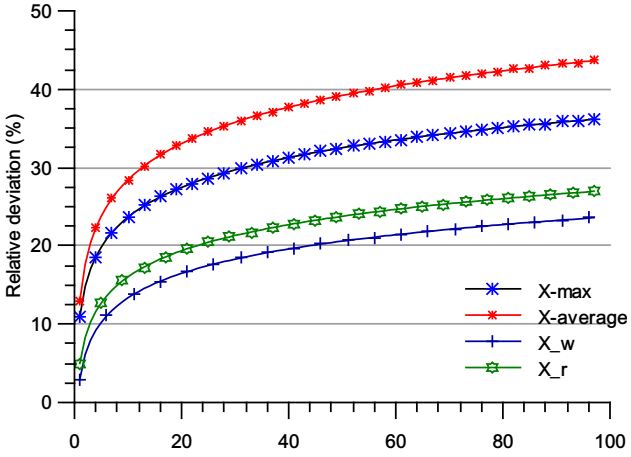
FIGURE 5.6: Optimal robust solution worst-case makespans in the preemptive case: small size workshops under high uncertainty



a. (m=3,n=10,d=3)



b. (m=3,n=50,d=3)



c. (m=3,n=200,d=3)

m=3	n	y	R^2
X_{max}	10	11.23ln(x) + 44.31	0.98
	50	11.43ln(x) + 24.05	0.97
	200	5.5ln(x) +13.2	0.97
$X_{average}$	10	12.57ln(x) + 45.56	0.96
	50	12.13ln(x) + 24.44	0.97
	200	6.73ln(x) +13.08	0.96
X_w	10	13ln(x) + 12	0.96
	50	12ln(x) + 6	0.965
	200	4.5ln(x) + 3	0.96
X_r	10	13.16ln(x) + 15	0.960
	50	12.9ln(x) + 8.7	0.965
	200	4.8ln(x) + 5	0.96

d. logarithmic trends

small, the optimization in the deterministic case of the nominal scenario can lead to an important makespan value, which makes the relative deviation between this one and the worst-case makespan of the robust optimal solution very small. And consequently, it makes the robust solution cost very interesting.

5.3 Conclusion

In this chapter, we have used the robust optimization techniques to address the preemptive unrelated parallel machine scheduling under uncertain processing times. We have proposed robust formulations of the

($Rm|pmtn|C_{max}$) based on the linear formulation proposed by Lawler and Labetoulle (1978) in which we propose to compute robust optimal assignment solutions first. Secondly, for each scenario, we constructed schedules corresponding to these robust assignment solutions. These algorithms are interesting when the decision makers are interested in computing a unique assignment solution. In which concerns the sequence, they may tolerate to have different sequences and permutations (one per scenario). We used the same ratio technique as in Chapter4. Consequently, we obtained the following complexity results:

TABLE 5.5: Complexity results in the case of preemption

DmM(R)$(\alpha\|\beta\|\gamma;\theta)$	*Complexities*	*References*
DmM$(Rm\|Split\|C_{max};p_{ij})$	Polynomial	Section 5.1.1
DmMR$(Rm\|Split\|C_{max};p_{ij})$	Polynomial	Section 5.1.2

The computational results showed that the optimal robust solution that minimizes the worst-case makespan (respectively the maximal regret) improved the results and lead to robust schedules with acceptable robustness costs under uncertainty. They follow the same trend line as in the splitting case. However, their initial robustness costs are slightly superior but the improvement ratios compared to X_{max} and X_r are still interesting. To go further, we propose in the next chapter to evaluate the stability of the robust solutions when a new scenario that is not taken into account is faced. The objective is to show that the proposed robust solutions are robust stable outside the set of discrete scenarios that is defined.

Chapter 6

Stability analysis in unrelated parallel machines under uncertain processing times

Abstract: In this chapter, we develop a framework to support decision in unrelated parallel machine scheduling under uncertain processing times. The uncertainty arising from a range of plausible futures is anticipated through a set of potential scenarios as shown in the previous chapters. Alternative robust solutions are generated and ranked according to a robustness measure. But because the future could change, the stability of these robust solutions should be evaluated under new scenarios induced by variations. We define the stability of a robust solution as its ability to cover a new scenario with minor deviations in both the solution structure and the solution performance. We use classical and new stability measures to evaluate the structural stability of the robust solutions and we calculate the worst-case makespan increase to evaluate the performance stability. Our motivation is to provide a decision support to help the decision-maker to choose among the robust solutions those with the most stable structure and the most stable performance. We perform an experimental study of these different metrics and run the model that includes the alternative solutions many times to determine how these solutions perform in a range of uncertainty degree.

Contents

6.1 Introduction

To further develop the work presented in the previous chapters, we consider the fact that in manufacturing systems, discrete scenario representation permit to limit the set of possible futures but, in reality the workshop will often face new scenarios because of numerous factors of perturbations. Thus, focusing on the robustness as a unique objective can be restrictive. In fact, the solution robustness is guaranteed only for processing time realizations that belong to the represented scenario set. Consequently, from one hand, the robust solution might have no robustness guarantee if the real scenario is a new one. On the other hand, the robust approach is not useful to help the decision-maker understand how much the objective could deviate, nor how much adjustment should be taken to cover it. Indeed, depending on whether a plausible scenario is taken into account or not, the structure of the robust solution could be very different. This aspect of robustness dependency to the chosen scenarios was already pointed out by Roy (2010) who claims that "adding or removing a scenario from the set of scenarios may lead to defining very different solutions as robust". From practitioners viewpoint, the quantification of these deviations under new scenarios both in terms of structure and performance is a very important question. Moreover, schedulers confronted with data uncertainty agree that it is difficult to list all the potential scenarios. Therefore, it is crucial to identify robust solutions whose structure and performance do not deviate significantly under new scenarios or to be able to choose the desired trade-off. In other words, it is important to choose among the robust solutions those with the most stable structure and the most stable performance.

In Chapters 3, 4 and 5, we dealt with the uncertain processing times in unrelated parallel machine scheduling under splitting (*resp.* preemption) by providing robust solutions under discrete processing time scenarios. The proposed approaches provided an efficient way to incorporate uncertainty in the models and to provide robust solutions. But, for both practical and computational reasons, it is impossible to take into account all the uncertainties. Demand uncertainty that induces uncertain processing times results from a variety of reasons: seasonal effect, changes in customers interests and needs, the number and the product quality/price of the competitors, *etc.* Consequently, it could be difficult for some sector of activity or during some economic periods to list all the potential scenarios, which means that a new scenario is always susceptible to occur. As the structure and the performance of the robust solution are very dependent on the set of discrete scenarios, the stability of the solutions can be affected if the real scenario is a not captured by the scenario generation process. Therefore, we evaluate the stability of the structure and the performance of the robust solutions when adding new scenarios. In this approach, the term stability is used for the algorithm step at which robust solutions of the problem have been already computed, and additional calculations are performed in order to investigate how these solution structure and performance depend on changes in the set of potential scenarios. In Section 6.2, we describe a robust solutions generation and stability analysis approach. In Section 6.3, we make a statistical analysis of the results. We choose the splitting problem as a case study to apply the approach and we show accordingly that the robust solution computed in Chapters 3, 4 are globally stable.

6.2 Robust solution stability analysis approach

The robust solution stability analysis approach is mainly based on a 2-stage algorithm (see Algorithm 4). The first stage is the Robust Solution Generation (RSG) module while the second stage (or step) is the Robust Solution Stability Evaluation (RSSE) module. We suppose that a scenario of reference always exists, we call this scenario: the nominal scenario. The algorithm starts with this nominal scenario $s_{nominal}$, the set of discrete scenarios is initially reduced to a single element: Ω_k=$\{s_{nominal}\}$. We set up the deterministic problem $(Rm/Split/C_{max})$ with the nominal scenario $s_{nominal}$ and compute its optimal solution denoted as X^1_{robust}.

Algorithm 4: Robust solution stability analysis algorithm

Data: $s_{nominal}$ the nominal scenario
K: the number of iterations
Generate a nominal scenario;
Set up the deterministic problem based on the nominal scenario $s_{nominal}$;
$k \leftarrow 1$;
$s_k \leftarrow s_{nominal}$;
$\Omega_k \leftarrow \{s_k\}$;
repeat
  $k \leftarrow k + 1$;
  Generate a new potential s_k ;
  Increase the size of potential scenario set Ω_k ;
  $\Omega_k \leftarrow \Omega_{k-1} \cup s_k$;
  Generate robust solutions for the set Ω_k through RSG module;
  Evaluate the stability of robust solutions through RSSE module;
until $k = K$;
analyse the trade-off robustness stability of each robust solution;

The module RSG is designed to compute a set of robust solutions and their worst-case makespan at each iteration. For this purpose, at each iteration k, we generate a new potential scenario s_k that is a variation around the nominal scenario and we add it to the set of potential scenarios: Ω_k= $\Omega_{k-1} \cup \{s_k\}$. Then, we calculate a set of robust solutions. We call X^k_{robust} the robust solution that considers the k scenarios that belong to Ω_k. RSG uses all the approaches developed in the previous Chapters (3,4 and 5). It computes the set of robust solutions: X^k_{max}, $X^k_{average}$, X^k_{median}, X^k_{min}, X^k_w and X^k_r and it computes their corresponding worst-case makespans over Ω_k.

In the same iteration k, RSSE module is designed to evaluate the stability of each robust solution when we confront a new potential scenario *i.e*, it measures the deviation between X^{k-1}_{robust} and X^k_{robust}. For this purpose, we evaluate the stability of each robust solution according to two components: the stability of its performance and the stability of its structure.

The iterations go on until the stopping criterion is satisfied: a stability threshold fixed by the decision-maker is reached by a robust solution or the variations of the stability measures over the iterations become negligible. Once, a robust solution is satisfying the stability objectives, Ω_K could be defined as the minimal

scenario set to consider in order to compute a robust stable solution: "Ω_K is the best uncertainty-immunized scenario set" that can be associated to our uncertain problem.

In the last step of the algorithm, the robustness and stability measures computed over the iteration permit to compare the different robust solutions and allow the analysis of the trade-off that offers each robust solution. We will use visualization and statistical analysis of the resulting runs to help decision-makers distinguish solutions that perform well from those that perform poorly. These informations can help decision-makers identify, evaluate, and choose robust strategies that resist over a wide range of futures and that are most stable under changes.

6.2.1 Robust Solution Generation

In the Robust Solution Generation module (RSG), at each iteration k, we update the set of potential scenarios Ω_k and we compute over Ω_k the optimal robust solutions (X_w^k and X_r^k) and the robust solutions based on the artificial scenario approach (X_{max}^k, $X_{average}^k$, X_{median}^k and X_{min}^k). Then, we measure the performance stability and structure stability of each solution.

6.2.2 Stability Analysis of robust solution performances

Algorithm 5: Robust Solution Generation algorithm at iteration k

1 Compute a robust solution X_{robust}^{k-1} for Ω_{k-1} ;
2 Compute the local performances of X_{robust}^{k-1} when applied to Ω_{k-1} ;
3 Compute and save the robustness measures of X_{robust}^{k-1} when applied to Ω_{k-1} ;
4 Update the set of potential scenarios by adding s_k: Ω_k=$\Omega_{k-1} \cup s_k$;
5 Compute the local performances of X_{robust}^{k-1} when applied to s_k ;
6 Compute and save the robustness measures of X_{robust}^{k-1} when applied to Ω_k ;
7 Compare the robustness measures of X_{robust}^{k-1} for the two successive sets ;

At each iteration k, to measure the performance stability of a robust solution X_{robust}, we quantify the deviations of the robustness measure of each robust solution X_{robust}^{k-1} when we consider a new potential scenario (see Algorithm 5). For this purpose, for each robust solution X_{robust}^{k-1}, we calculate the difference between the worst-case makespan of X_{robust}^{k-1} when applied to the set of scenarios Ω_{k-1} and its worst-case makespan when applied to the set of scenarios Ω_k. We call this difference the worst-case makespan deviation at iteration k and we calculate it as:

$$WCMD^k(X_{robust}) = \max_{s\in\Omega_k} \max_{i\in M} \sum_{j=1}^{n} p_{ij}^s x_{ij}^{k-1} - \max_{s\in\Omega_{k-1}} \max_{i\in M} \sum_{j=1}^{n} p_{ij}^s x_{ij}^{k-1}$$

6.2.3 Stability analysis of robust solution structures

When adding a new potential scenario, the instability of solutions in parallel machine results primarily in machine assignment. At each iteration k, to measure the structure stability of a robust assignment solution X_{robust}, we compare X^k_{robust} and X^{k-1}_{robust} by computing:

- the number of perturbed jobs $PJ^k_j(X_{robust})$ (see Algorithm 7);
- the number of perturbed machines $PM^k_i(X_{robust})$ (see Algorithm 8);
- the number of changing ratios $CR^k(X_{robust})$ (see Algorithm 6):
- the number of new assignments $NA^k(X_{robust})$ (see Algorithm 9);
- the number of cancelled assignments $CA^k(X_{robust})$ (see Algorithm 10);
- the magnitude of ratio changes $MC^k(X_{robust})$ (see Algorithm 11).

Algorithm 6: The number of changing ratios computation at iteration k

Data: $M = \{1, \ldots, m\}$ the set of m machines, $N = \{1, \ldots, n\}$ the set of n jobs
$X^{k-1}_{robust} = (x^{k-1}_{ij})_{i,j}$ the robust solution for Ω_{k-1}
$X^k_{robust} = (x^k_{ij})_{i,j}$ the robust solution for Ω_k
Result:
$CR^k(X_{robust})$: number of changed ratio in the robust solution X_{robust} at iteration k.
// Initialisation
$CR^k(X_{robust}) \leftarrow 0$;
foreach $i \in M$ **do**
  foreach $j \in N$ **do**
    if $(x^k_{ij} \neq x^{k-1}_{ij})$ **then**
      $CR^k(X_{robust}) \leftarrow CR^k(X_{robust}) + 1$;
    end
  end
end
return $CR^k(X_{robust})$

$CR^k(X_{robust})$ measures the number of ratios that have increased or decreased when we consider a new potential scenario. This stability measure gives a macroscopic idea about the structural similarities between the robust solution X^k_{robust} that considers a new potential scenario and the robust solution X^{k-1}_{robust} that does not:

$$CR^k_{ij}(X_{robust}) = \begin{cases} 1 & \text{if } j \text{ is assigned to } i \text{ with a different ratio in } X^k_{robust} \text{ compared to } X^{k-1}_{robust} \\ 0 & \text{otherwise} \end{cases}$$

$$CR^k(X_{robust}) = \sum_{i \in M} \sum_{j \in N} CR^k_{ij}(X_{robust})$$

Algorithm 7: The number of perturbed jobs computation at iteration k

Data: $M = \{1, \ldots, m\}$ the set of m machines, $N = \{1, \ldots, n\}$ the set of n jobs
$X^{k-1}_{robust} = (x^{k-1}_{ij})_{i,j}$ the robust solution for Ω_{k-1}
$X^{k}_{robust} = (x^{k}_{ij})_{i,j}$ the robust solution for Ω_{k}
Result:
$PJ^k(X_{robust})$: number of perturbed jobs in the robust solution X_{robust} at iteration k.
// Initialisation
$PJ^k(X_{robust}) \leftarrow 0$;
foreach $j \in N$ **do**
  $PJ^k_j(X_{robust}) \leftarrow 0$;
  $i \leftarrow 1$;
  while *(*$i < m$ *and* $PJ^k_j(X_{robust}) = 0$*)* **do**
    if $(x^k_{ij} \neq x^{k-1}_{ij})$ **then**
      $PJ^k_j(X_{robust}) \leftarrow 1$;
      $PJ^k(X_{robust}) \leftarrow PJ^k(X_{robust})$+1;
    end
    $i \leftarrow i$+1 ;
  end
end
return $PJ^k(X_{robust})$

Algorithm 8: The number of perturbed machines computation at iteration k

Data: $M = \{1, \ldots, m\}$ the set of m machines, $N = \{1, \ldots, n\}$ the set of n jobs
$X^{k-1}_{robust} = (x^{k-1}_{ij})_{i,j}$ the robust solution for Ω_{k-1}
$X^{k}_{robust} = (x^{k}_{ij})_{i,j}$ the robust solution for Ω_{k}
Result:
$PM^k(X_{robust})$: number of perturbed machines in the robust solution X_{robust} at iteration k.
// Initialisation
$PM^k(X_{robust}) \leftarrow 0$;
foreach $i \in M$ **do**
  $PM^k_i(X_{robust}) \leftarrow 0$;
  $j \leftarrow 1$;
  while *(*$j < n$ *and* $PM^k_i(X_{robust}) = 0$*)* **do**
    if $(x^k_{ij} \neq x^{k-1}_{ij})$ **then**
      $PM^k_i(X_{robust}) \leftarrow 1$;
      $PM^k(X_{robust}) \leftarrow PM^k(X_{robust})$+1;
    end
    $j \leftarrow j$+1 ;
  end
end
return $PM^k(X_{robust})$

Nevertheless, $CR^k(X_{robust})$ does not quantify how many jobs and how many machines are involved in this ratio change. Therefore, we measure the number of disrupted jobs PJ^k and the number of disrupted machines PM^k. PJ^k=$\sum_{j\in N} PJ_j^k$ where

$$PJ_j^k(X_{robust}) = \begin{cases} 1 & \text{if job } j \text{ is disrupted when we take into account a new potential scenario at iteration } k \\ 0 & \text{otherwise} \end{cases}$$

and PM^k=$\sum_{i\in M} PM_i^k$ where

$$PM_i^k(X_{robust}) = \begin{cases} 1 & \text{if machine } i \text{ is disrupted when we take into account a new potential scenario at iteration } k \\ 0 & \text{otherwise} \end{cases}$$

Algorithm 9: The number of new assignments assignments computation at iteration k

Data: $M = \{1, \ldots, m\}$ the set of m machines, $N = \{1, \ldots, n\}$ the set of n jobs
$X_{robust}^{k-1} = (x_{ij}^{k-1})_{i,j}$ the robust solution for Ω_{k-1}
$X_{robust}^{k} = (x_{ij}^{k})_{i,j}$ the robust solution for Ω_{k}
Result:
$NA^k(X_{robust})$: number of new assignments in the robust solution X_{robust} at iteration k.
// Initialisation
$NA^k(X_{robust}) \leftarrow 0$;
foreach $i \in M$ **do**
  foreach $j \in N$ **do**
    $NA_{ij}^k(X_{robust}) \leftarrow 0$;
    if (x_{ij}^k>0 *and* $x_{ij}^{k-1} = 0$) **then**
      $NA_{ij}^k(X_{robust}) \leftarrow 1$;
      $NA^k(X_{robust}) \leftarrow NA^k(X_{robust}) + 1$;
    end
  end
end
return $NA^k(X_{robust})$

To measure extreme ratio changes, we introduce $NA^k(X_{robust})$ and $CA^k(X_{robust})$. $NA^k(X_{robust})$ measures the number of ratios that were null in X_{robust}^{k-1} and takes a positive value in X_{robust}^{k} when we consider a new potential scenario. It reflects the number of additional setups or changeovers in machines when we take into account a new potential scenario. Similarly, $CA^k(X_{robust})$ measures the number of ratios that were positive in X_{robust}^{k-1} and then become null in X_{robust}^{k}. It may report a machine setup that is no longer necessary when we consider a new potential scenario, and may be behind this cancellation, another machine has to take the job in order to make up the difference. $NA^k(X_{robust})$ and $CA^k(X_{robust})$ are computed such

Algorithm 10: The number of cancelled assignments computation at iteration k

Data: $M = \{1, \ldots, m\}$ the set of m machines, $N = \{1, \ldots, n\}$ the set of n jobs
$X^{k-1}_{robust} = (x^{k-1}_{ij})_{i,j}$ the robust solution for Ω_{k-1}
$X^{k}_{robust} = (x^{k}_{ij})_{i,j}$ the robust solution for Ω_{k}
Result:
$CA^k(X_{robust})$: number of cancelled assignments in the robust solution X_{robust} at iteration k.
// Initialisation
$CA^k(X_{robust}) \leftarrow 0$;
foreach $i \in M$ **do**
  foreach $j \in N$ **do**
    $CA^k_{ij}(X_{robust}) \leftarrow 0$;
    if (x^k_{ij}=0 *and* $x^{k-1}_{ij} > 0$) **then**
      $CA^k_{ij}(X_{robust}) \leftarrow 1$;
      $CA^k(X_{robust}) \leftarrow CA^k(X_{robust}) + 1$;
    end
  end
end
return $CA^k(X_{robust})$

as:

$$NA^k_{ij}(X_{robust}) = \begin{cases} 1 & \text{if } x^{k-1}_{ij} = 0 \text{ and } x^k_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

and

$$CA^k_{ij}(X_{robust}) = \begin{cases} 1 & \text{if } x^{k-1}_{ij} > 0 \text{ and } x^k_{ij} = 0 \\ 0 & \text{otherwise} \end{cases}$$

Then:

$$NA^k(X_{robust}) = \sum_{i \in M} \sum_{j \in N} NA^k_{ij}(X_{robust}) \quad and \quad CA^k(X_{robust}) = \sum_{i \in M} \sum_{j \in N} CA^k_{ij}(X_{robust})$$

They are critical if a machine setup or tooling are prepared in advance, and jobs wait in a sequence queue based on the previous solution. They indicate a sequence deviation that will incur costs in handling and reallocating the jobs, and re-planning the tools changeover.

In relation with the $CR^k(X_{robust})$, $MC^k(X_{robust})$ measures the magnitude of ratio changes.

$$MC^k(X_{robust}) = \sum_{i \in M} \sum_{j \in N} sup(0, x^k_{ij} - x^{k-1}_{ij}) \tag{6.1}$$

Algorithm 11: The magnitude of ratio changes computation at iteration k

Data: $M = \{1, \ldots, m\}$ the set of m machines, $N = \{1, \ldots, n\}$ the set of n jobs
$X^{k-1}_{robust} = (x^{k-1}_{ij})_{i,j}$ the robust solution for Ω_{k-1}
$X^{k}_{robust} = (x^{k}_{ij})_{i,j}$ the robust solution for Ω_{k}
Result:
$MC^k(X_{robust})$: magnitude of ratio changes in the robust solution X_{robust} at iteration k.
// Initialisation
$MC^k(X_{robust}) \leftarrow 0$;
foreach $i \in M$ **do**
  foreach $j \in N$ **do**
    if $(x^k_{ij} - x^{k-1}_{ij} > 0)$ **then**
      $MC^k(X_{robust}) \leftarrow MC^k(X_{robust}) + x^k_{ij} - x^{k-1}_{ij}$;
    end
  end
end
return $MC^k(X_{robust})$

This measure gives an idea about the handling operations necessary to make the adjustment. We choose the sup operator to count the change only once.

In the cases where permutation decisions are also important, we can consider also the permutation deviation. But we do not consider this issue in this work.

Example 6.2.1 *Let us consider the same example used in the previous chapters. We add a new scenario s_7 to the set of six scenarios: We take the example of X_r.*

TABLE 6.1: Job processing requirements scenarios

	p_1	p_2	p_3	p_4
s_1	30	50	30	100
s_2	50	30	80	50
s_3	40	50	100	80
s_4	35	40	50	70
s_5	40	60	100	50
s_6	60	50	80	70
s_7	100	50	90	40

The new robust optimal solution X^7_r under the set ω_7 computed according to ($LP_{r-Split}$) is given as:

When we apply X^6_r to the scenario $s7$, we obtain a makespan that is equal to 30.14 units. The worst-case makespan of X^6_r has increased and it is equal to 30.14 units instead of 29.34 units. Thus, the $WCMD^7$ of X_r at iteration 7 is equal to 0.8 units.

TABLE 6.2: The optimal robust solution minimizing the maximal regret X_r in the splitting case under 7 scenarios

	J_1	J_2	J_3	J_4
M_1	0.29	0.15	0.60	0.08
M_2	0.5	0	0.17	0.41
M_3	0.21	0.85	0.22	0.51

TABLE 6.3: The optimal robust solution minimizing the maximal regret X_r in the splitting case under 6 scenarios

	J_1	J_2	J_3	J_4
M_1	0.57	0.25	0.47	0
M_2	0.43	0	0.21	0.40
M_3	0	0.75	0.32	0.6

Concerning the structure stability, we observe by comparing Table 6.2 and Table 6.3 that the solution structures are different. According to the structure stability metrics, we have the following results:

- a new assignment ($NA^7 = 1$).

- a cancelled assignments ($CA^7 = 1$).

- 11 changed ratios ($CR^7 = 11$).

- 4 perturbed jobs ($PJ^7 = 4$),

- 3 perturbed machines($PM^7 = 3$),

- 60% is the total magnitude of assignment ratio change ($MC^7 = 0.6$).

In the next section, we represent the computational results concerning the stability analysis of the robust solutions.

6.3 Computational results in the case of splitting: Statistical analysis

We consider the same data generator used along this thesis.

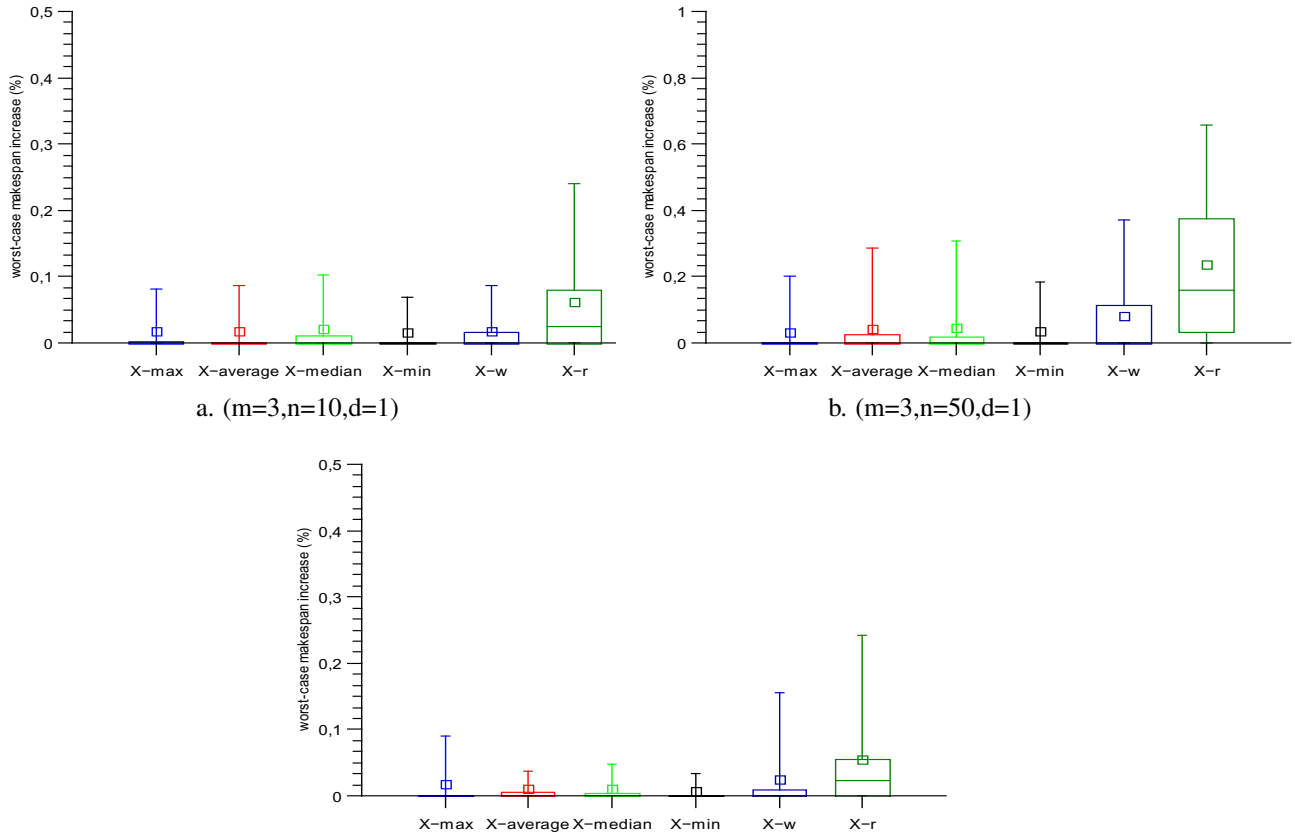
We fixed the number of iterations at K=100. At each iteration we compute the robust approximate solutions (X_{max}, X_{min}, $X_{average}$ and X_{median}) and the robust optimal solutions (X_w and X_r) in the splitting case. Then we evalute the stability of each solution according to all the stability indicators over the 100 iterations.

For each indicator, we represent the computational results through box-plots to represent the mean values and the distribution of the values around the mean value.

6.3.1 Performance stability evaluation

In this section, we evaluate the performance stability of the robust solutions. For this purpose, we report the computational results concerning the worst-case makespan deviation (WCMD) under new scenarios.

FIGURE 6.1: The worst-case makespan deviation under low uncertainty degree in the splitting case



a. (m=3,n=10,d=1)

b. (m=3,n=50,d=1)

c. (m=7,n=10,d=1)

Under low uncertainty degree, the robust solution worst-case makespan deviations are insignificant ***(*** $<< 1\%$***).***

In small size workshops under small job variety (Figure 6.1 graphic a.), the robust solution's WCMD are very low for all the solutions. The WCMD mean values of X_{max}, X_{min}, X_w, $X_{average}$ and X_{median} are around 0.01% with standard deviations that are around 0.03%. For X_r, the WCMD mean value is equal to 0.06% with a standard deviation that is equal to 0.08%.

When we increase the number of jobs, we remark that the WCMD does not increase except for X_w ***and*** X_r ***(e.g. Figure 6.1 graphic b.).*** X_w's WCMD mean value is equal to 0.07% with a standard deviation that is equal to 0.13%. And the WCMD mean value of X_r is equal to 0.23% with a standard deviation that is equal to 0.23%.

When we increase the numbers of machines, we notice a small decrease in the WCMD especially for X_r ***and*** X_w***.*** For instance, in medium size workshops under small jobs variety (Figure 6.1 graphic c.). X_w

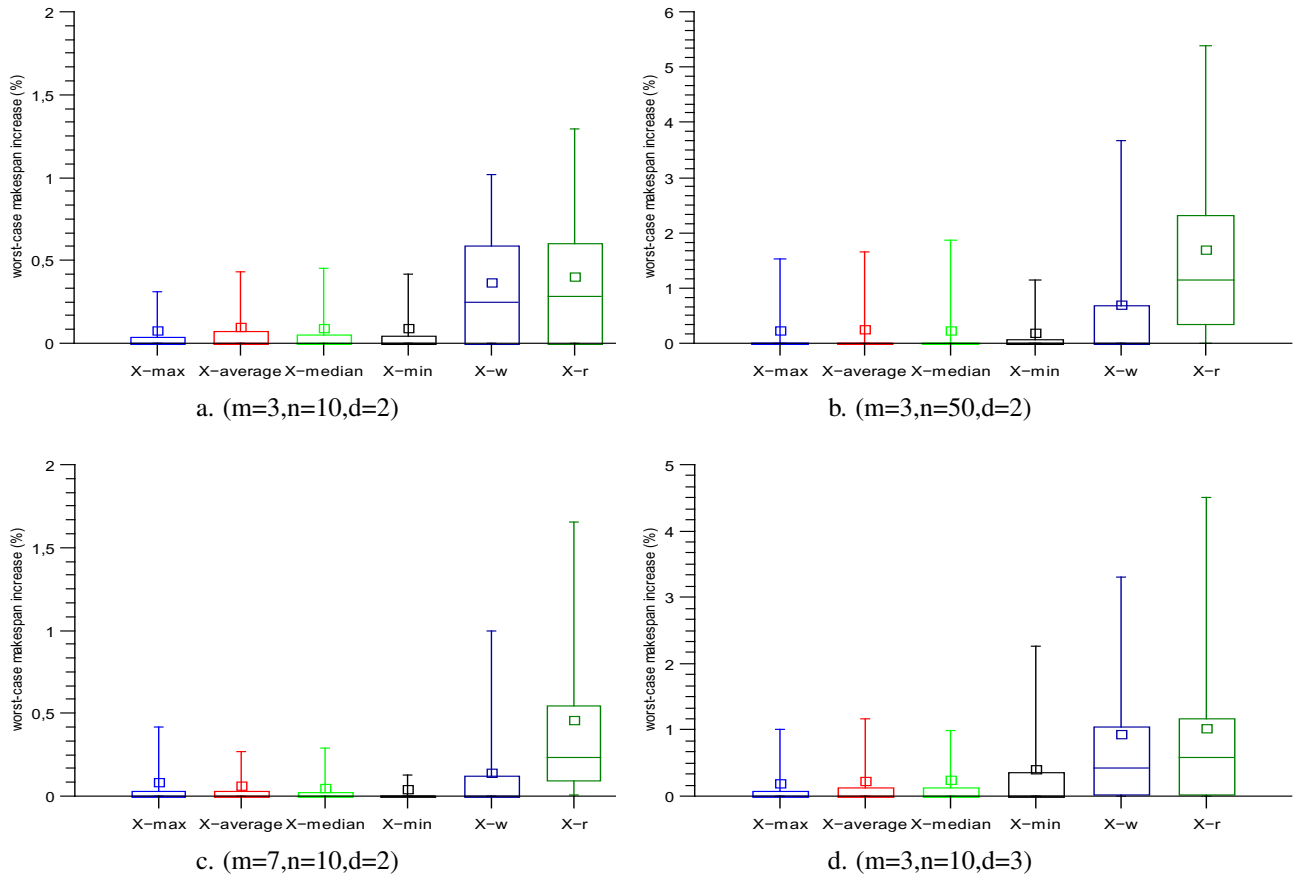
and X_r's WCMD mean values are respectively equal to 0.02% and 0.05% with standard deviations that are respectively equal to 0.05% and 0.08%.

When we increase the degree of uncertainty, we observe that the $WCMD$ ***of the solutions increase but their values are small.***

Under medium uncertainty degree, in small size workshops under low job variety (Figure 6.2 graphic a.), the increase is more important for X_w and X_r than for the other solutions. But the $WCMD$ is still very low for all the solutions. For X_{max}, X_{min}, $X_{average}$ and X_{median}, The $WCMD$ mean values are around 0.07% with standard deviations that are around 0.2%. And the $WCMD$ of X_w and X_r mean values are respectively equal to 0.35% and 0.38% with a standard deviation that is equal to 0.4%.

Under high degree of uncertainty, we observe that the WCMD mean value of X_{min}***,*** X_w ***and*** X_r ***become more significant relatively to the previous cases.*** For instance, in the case of small size workshops under medium job variety (Figure 6.2 graphic a.), for X_{min}, the WCMD mean value is equal to 1.2% with a standard deviation that is equal to 2.3%. $X_w$and X_r WCMD mean values are respectively equal to 1.9% and 2.6% with standard deviations that are respectively equal to 2.3% and 2.5%.

FIGURE 6.2: Increase of the worst-case makespan under medium and high uncertainty in different size workshops in the splitting case



a. (m=3,n=10,d=2)

b. (m=3,n=50,d=2)

c. (m=7,n=10,d=2)

d. (m=3,n=10,d=3)

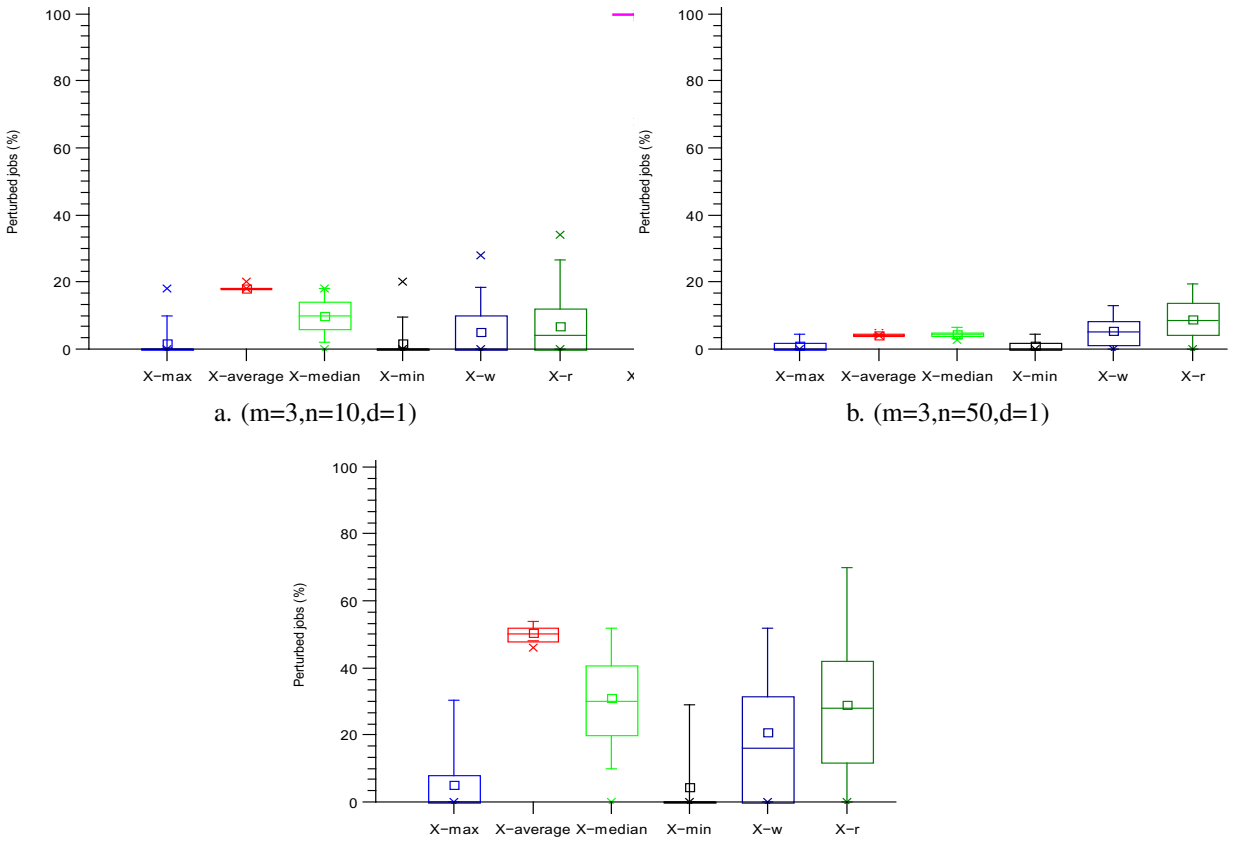
In the next sections, we report the results concerning the structure stability of the solutions.

6.3.2 Structure stability evaluation

In this section, we evaluate the structure stability of the solutions according to different structure stability indicators.

a- The number of perturbed jobs:

FIGURE 6.3: Number of perturbed jobs under low uncertainty degree in the splitting case



a. (m=3,n=10,d=1)

b. (m=3,n=50,d=1)

c. (m=7,n=10,d=1)

According to the PJs, under low uncertainty degree, X_{max} and X_{min} are the most stable.

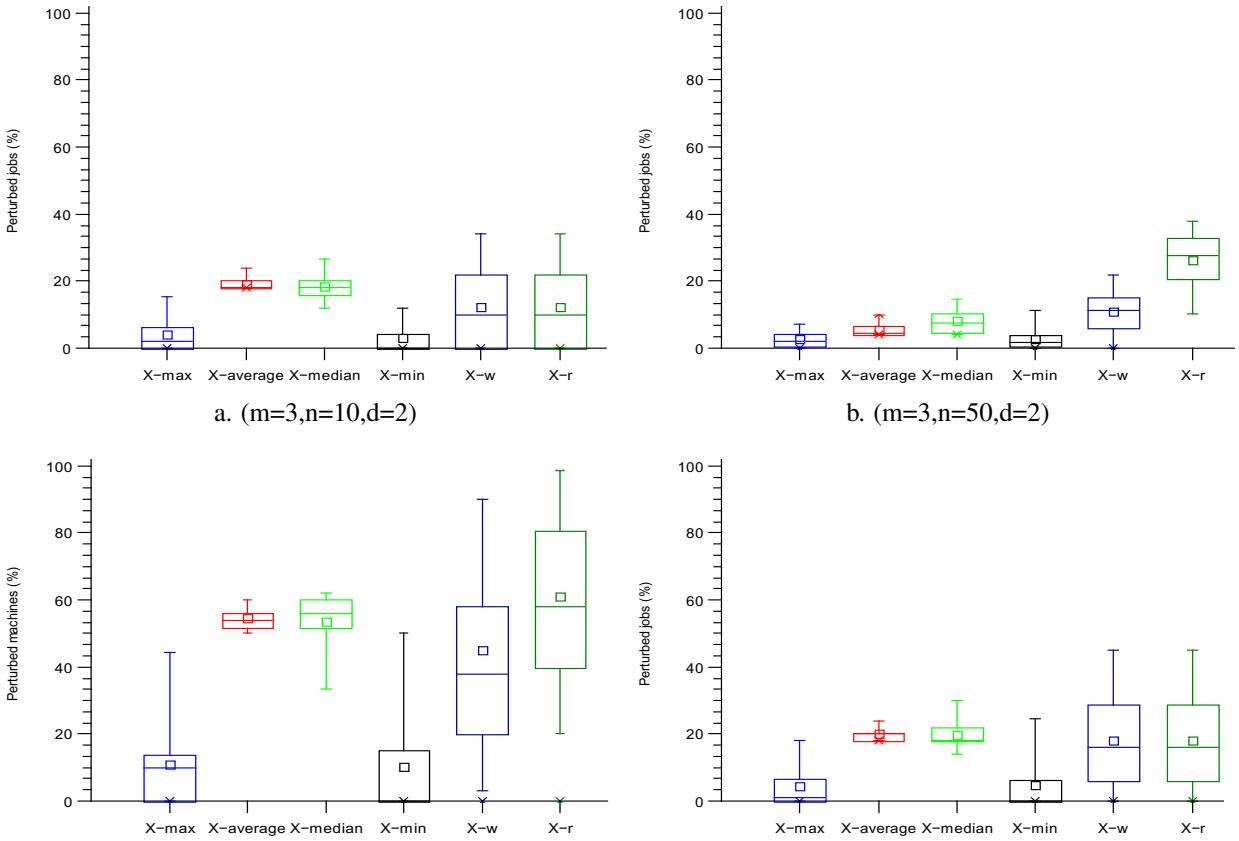
In small size workshops under small job variety (Figure 6.3 graphic a.), the PJs are low for all the solutions except for $X_{average}$ and X_{median}. X_{max} and X_{min} are the most stable, followed by X_w, X_r. PJ mean values of for X_{max} and X_{min} are respectively equal to 1.65% and 1.60% with standard deviations that are respectively equal to 3.63% and 3.98%. For X_w, PJ mean value is equal to 5.22% with a standard deviation that is equal to 7.06% and for X_r, the mean value of PJ is equal to 6.66% with a standard deviation that is equal to 8.75%. PJ become significant under X_{median}, PJ mean value is equal to 9.97% with a standard deviation that is equal to 4.64%. For $X_{average}$, PJ mean value increases highly, it is equal to 18.08% with a small standard deviation that is equal to 0.4%.

When we increase the number of jobs, we remark that the PJs decrease. X_{max} and X_{min} are the most stable, followed by $X_{average}$ and X_{median} instead of X_w and X_r. Under medium job variety (Figure 6.3

graphic b.), PJ mean values of X_{max} and X_{min} are respectively equal to 1.12% and 1.01% with a standard deviation that are respectively equal to 1.75% and 1.83%. For $X_{average}$ and X_{median}, PJ mean values are around 4.2% with standard deviations that are respectively equal to 0.46% and 1.05%. X_w's PJ mean value is equal to 5.4% with a standard deviation that is equal to 4.41%. And PJ mean value under X_r is more significant, it is equal to 8.8% with a standard deviation that is equal to 6.10%.

When we increase the number of machines, we remark that the PJs highly increase (both the mean values and the standard deviations). For instance, under medium workshop size with low job variety (Figure 6.3 graphic a.). For X_{min} and X_{max} which are the most stable, PJ mean value of X_{min} is equal to 4.33% with a standard deviation that is equal to 9.88% and PJ mean value of X_{max} is equal to 5.12% with a standard deviation that is equal to 11.05%. For X_w, PJ mean value is equal to 20.79% with a standard deviation that is equal to 18.86%. For X_r, PJ mean value is equal to 29.1% with a standard deviation that is equal to 21.32%. For X_{median}, PJ mean value is equal to 31.08% with a standard deviation that is equal to 13.22%. And for $X_average$ which is the less stable, PJ mean value is very high, it is equal to 50.62% with a standard deviation that is equal to 2.38%.

FIGURE 6.4: Number of perturbed jobs under medium and high uncertainty degrees in the splitting case



a. (m=3,n=10,d=2)

b. (m=3,n=50,d=2)

c. (m=7,n=10,d=2)

d. (m=3,n=10,d=3)

When we increase the degree of uncertainty, the PJs increase also. The increase in the PJs is more

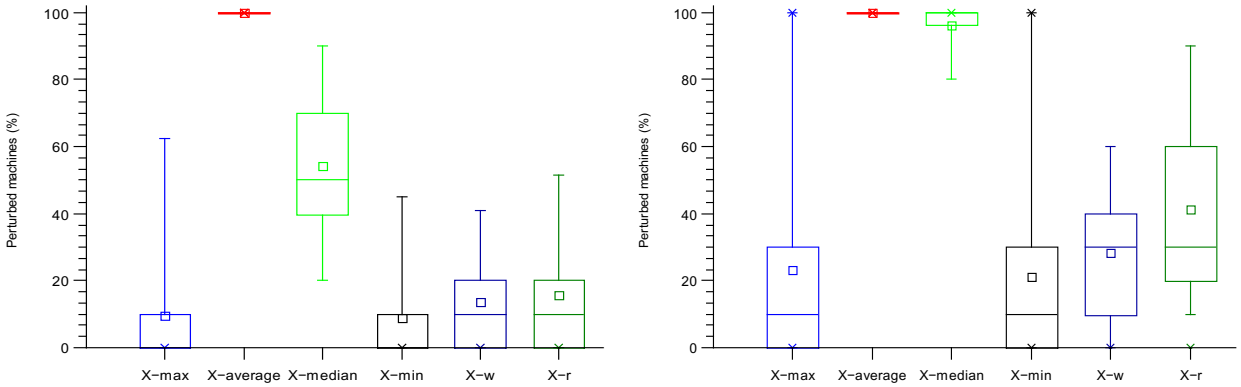
important when the number of machines increase. For instance, ***under medium uncertainty degree***, in medium workshop size and low job variety (Figure 6.4 graphic c.), only X_{max} and X_{min} are relatively stable as their PJ mean values are less than 11% while all the other solutions are unstable as their mean values of perturbed jobs are superior to 50%. The PJ mean value of X_{max} is equal to 11.08% with a standard deviation that is equal to 14.71%. For X_{min}, the PJ mean value is equal to 10.22% with a standard deviation that is equal to 15.06%. For X_w, thePJ mean value is equal to 44.90% with a standard deviation that is equal to 25.39%. For X_r, the PJ mean value is equal to 61% with a standard deviation that is equal to 26.60%. For X_{median}, the mean value of perturbed jobs is equal to 53.5% with a standard deviation that is equal to 20.60%. And For $X_{average}$, the PJ mean value is equal to 54.66% with a standard deviation that is equal to 2.95%.

Under high uncertainty degree, we observe that the PJs have the same properties and the same magnitude as in the case of medium uncertainty (see for instance Figure 6.4).

b- The number of perturbed machines:

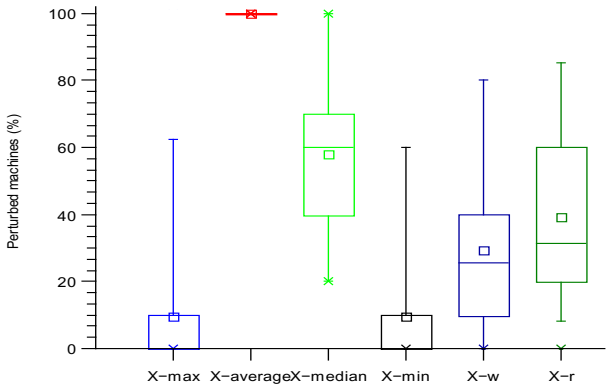
In which concerns the number of perturbed machines (PM), the computational results show the following properties: ***Under low uncertainty degree, X_{min} and X_{max} are the most stable regarding the PMs,***

FIGURE 6.5: Number of perturbed machines under low uncertainty degree in the splitting case



a. (m=3,n=10,d=1)

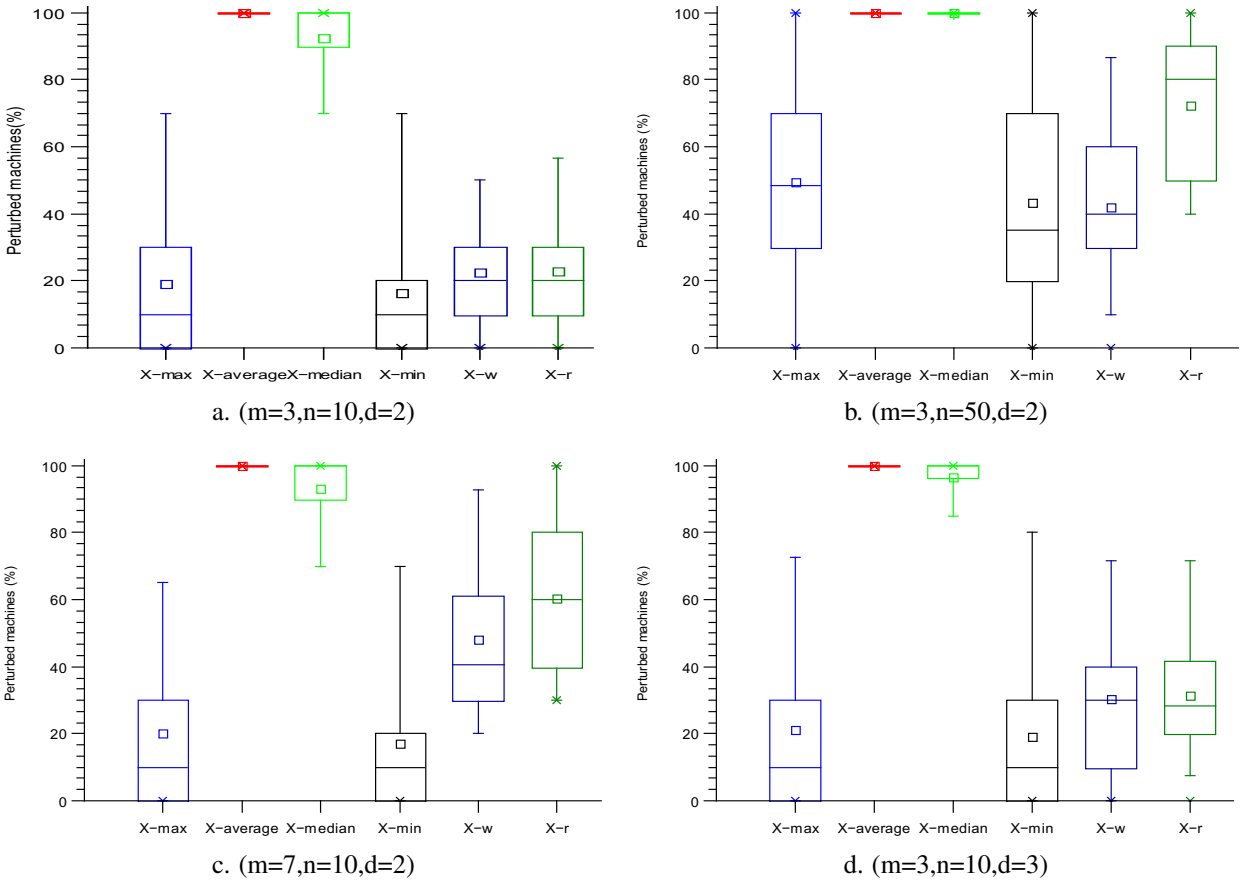
b. (m=3,n=50,d=1)



c. (m=7,n=10,d=1)

followed by X_w and X_r. The other solutions are not stable. In small size workshops under small job variety (Figure 6.5 graphic a.), The PM mean values under X_{min} and X_{max} do not exceed 10% but their standard deviations are important. X_w and X_r's PM mean values are around 13% with relatively important standard deviation. And X_{median} and $X_{average}$ are not stable. The PM mean value of X_{max} is equal to 9.47% with a standard deviation that is equal to 20.69%. For X_{min}, the PM mean value is equal to 8.81% with a standard deviation that is equal to 19%. X_w and X_r lead to similar results. For X_w, the PM mean value is equal to 13.5% with a standard deviation that is equal to 19.67%. For X_r, the PM mean value is equal to 15.76% with a standard deviation that is equal to 17.96%. For X_{median}, the PM mean value is equal to 55% with a standard deviation that is equal to 26%. $X_{average}$ is not stable, the PM mean value under $X_{average}$ is equal to 100%.

FIGURE 6.6: Number of perturbed machines under medium and high uncertainty degrees in the splitting case



a. (m=3,n=10,d=2) b. (m=3,n=50,d=2)

c. (m=7,n=10,d=2) d. (m=3,n=10,d=3)

According to the distributions, the PM upper values can reach 100% for all the solutions except for X_w and X_r for which they are respectively equal to 80% and 90% while the lower value is equal to 0%.

When we increase the number of jobs, we remark that the PMs increase. For instance, under medium job variety (Figure 6.5 graphic b.), the ranking of the solutions regarding the PMs is the same but both the mean values, the standard deviations and the upper values are more important. The PM mean value of

X_{max} is equal to 23.15% with a standard deviation that is equal to 33.58%. For X_{min}, the PM mean value is equal to 21.25% with a standard deviation that is equal to 31.40%. For X_w, the PM mean value is equal to 28.33% with a standard deviation that is equal to 17.9%. The PM mean value is more important under X_r and it is equal to 41.42% with a standard deviation that is equal to 26.6%. $X_{average}$, X_{median} are not stable, the PM are around 100%. All the PM upper values are reaching the 100%.

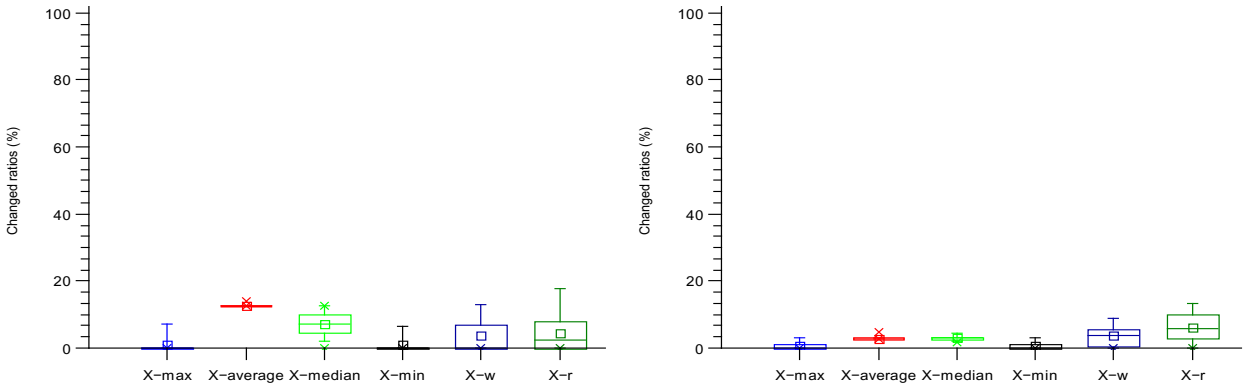
When we increase the number of machines, we remark that the PMs slightly increase for the solutions X_w, X_r. But they slightly decrease for X_{min} and X_{max}. For instance, under medium workshop size and low job variety (Figure 6.5 graphic c.). The PM mean value of X_{max} is equal to 9.68% with a standard deviation that is equal to 20.49%. Similar values are obtained under X_{min}. For X_w and X_r, the mean values of perturbed machines are respectively equal to 29% and 39% with a standard deviation that is around 25%. For X_{median}, the PM mean value is equal to 58.2% with a standard deviation that is equal to 22%. And the PM upper values are all superior to 90%.

When we increase the degree of uncertainty, the PMs highly increase for all the solutions. Under medium uncertainty degree, in small size workshops under low job variety (Figure 6.6 graphic a.), X_{min}, X_{max}, X_w and X_r are quite stable while X_{median} and $X_{average}$ are unstable according to the PMs. The PM mean value of X_{min} is equal to 16% with a standard deviation that is equal to 22%. X_{max}, X_w and X_r lead to similar results. Their PM mean values are around 21% with standard deviation that are around 20%. For X_{median} and $X_{average}$, the PM mean value are more than 92%. When we increase the number of jobs or machines, almost all the solutions are unstable regarding the PMs (*e.g.* Figure 6.6 graphics b. and c.).

Under high degree of uncertainty, the results are similar to the case of medium uncertainty. The increase is relatively small (Figure 6.6).

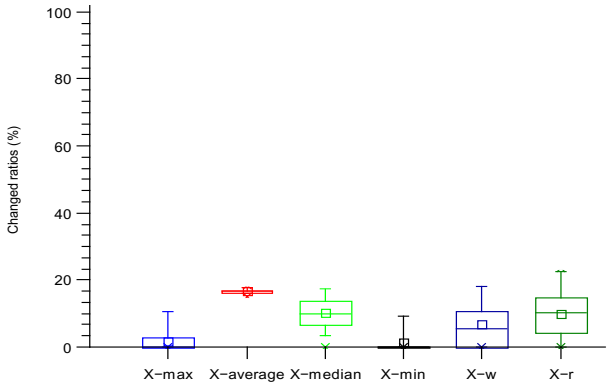
c- The changing ratios

FIGURE 6.7: Number of changing ratios under low uncertainty degree in the splitting case



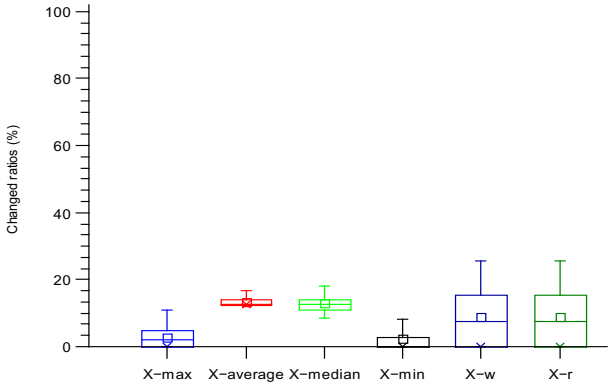
a. (m=3,n=10,d=1) b. (m=3,n=50,d=1)

In which concerns the number of changed ratios CRs, ***under low uncertainty degree, the CRs are very low. Furthermore, X_{max} and X_{min} are the most stable***. For example, in small size workshops under small job variety (Figure 6.7 graphic a.), the CRs are low for all the solutions except for $X_{average}$. X_{max} and X_{min} are the most stable, followed by X_w, X_r and X_{median}. The CR mean values of X_{max} and X_{min}

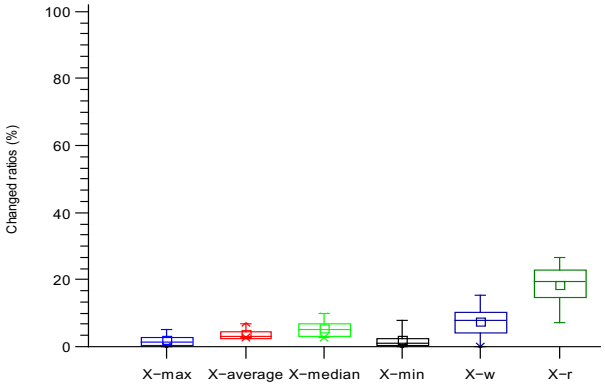


a. (m=7,n=10,d=1)

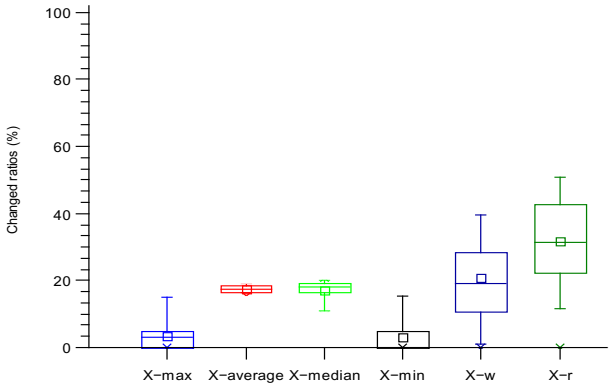
FIGURE 6.8: Number of changing ratios under medium and high uncertainty degrees in the splitting case



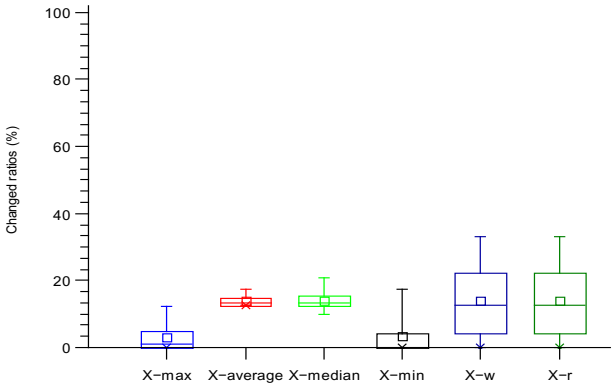
a. (m=3,n=10,d=2)



b. (m=3,n=50,d=2)



c. (m=7,n=10,d=2)



d. (m=3,n=10,d=3)

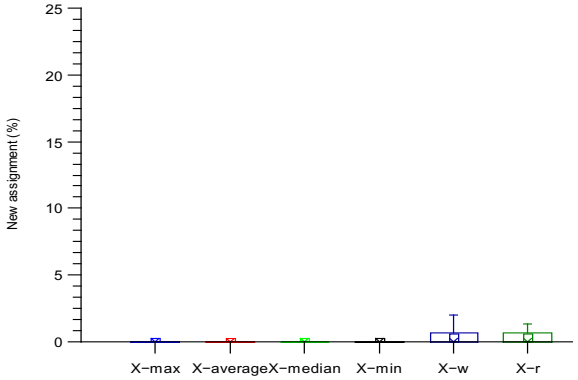
are respectively equal to 1.14% and 1.17% with standard deviations that are respectively equal to 2.54% and 2.78%. For X_w, the CR mean value is equal to 3.61% with a standard deviation that is equal to 4.85%. For X_r, the CR mean value is equal to 4.54% with a standard deviation that is equal to 5.96%. The CR mean value increases under $X_{average}$ and it is equal to 12.72% but, it standard deviation is equal to only 0.26%.

When we increase the number of jobs, we remark that the CRs decrease. X_{max} and X_{min} are the most stable, followed by $X_{average}$, X_{median} instead of X_w and X_r. And ***when we increase the number of machines, the increase of the CRs is not significant (see Figure 6.7 graphic c.).***

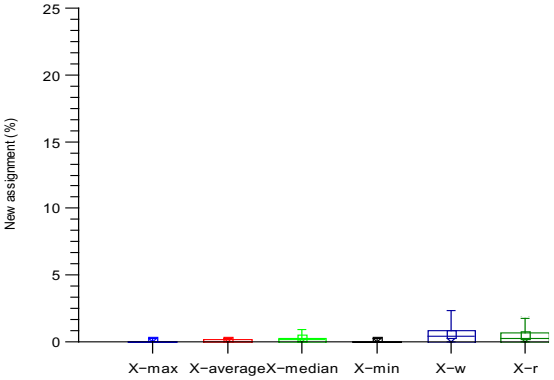
When we increase the degree of uncertainty, the CRs increase. However, under medium uncertainty degree, the increase is very small. In small size workshops under small job variety (Figure 6.8 graphic a.), the CR mean values are low for X_{max} and X_{min} (less than 3%). But, the CRs become significant under X_w, X_r, X_{median} and $X_{average}$. For X_w and X_r, the CR mean values are equal to 10% with a standard deviation that is equal to 8%. And for X_{median} and $X_{average}$, the CR mean values are around to 13% with standard deviations that are around 1.5%. ***Under high uncertainty degree***, the results are very similar to the case of medium uncertainty degree (see Figure 6.8).

d- The number of new assignments and cancelled assignments

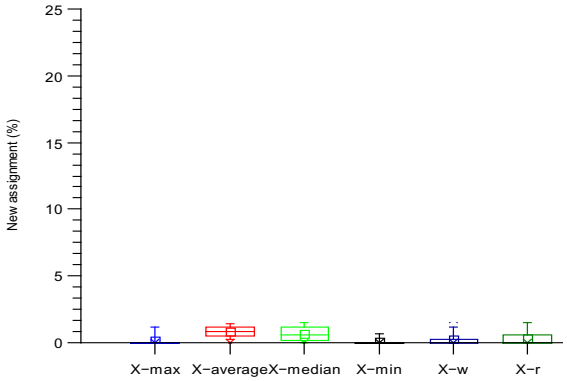
FIGURE 6.9: Number of new assignments under low uncertainty degree in the splitting case



a. (m=3,n=10,d=1)



b. (m=3,n=50,d=1)



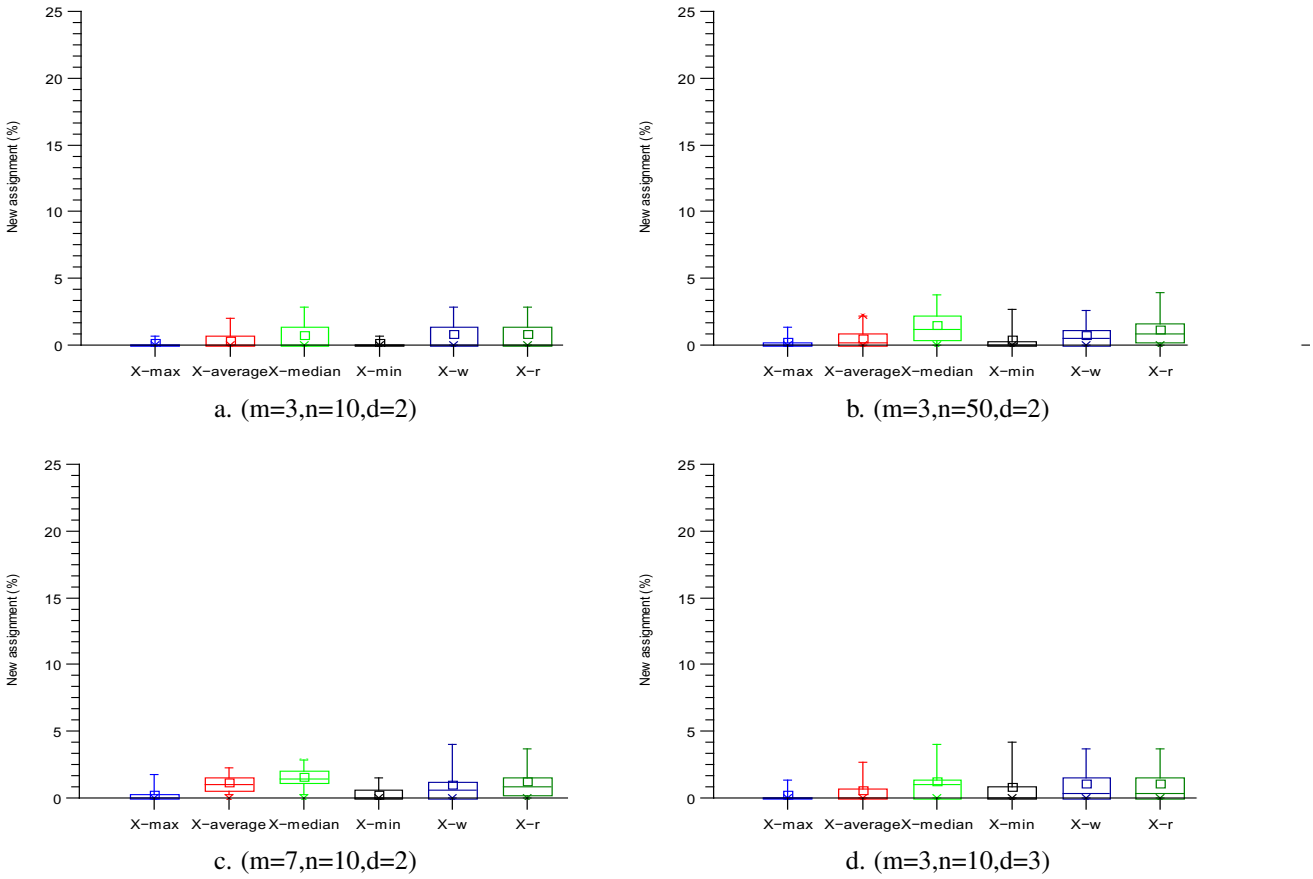
c. (m=7,n=10,d=1)

Under low uncertainty degree, the number of new assignments (NA) is very small under all the solutions (less than 1%). For instance, in small size workshops under small job variety (Figure 6.9 graphic a.), for X_{max}, X_{min}, X_{median} and $X_{average}$, the NA mean values are less than 0.05% and their standard deviations are less than 0.5%. For X_w and X_r, the NA mean values are around 0.5% and their standard deviations are around 0.5%.

When we increase the number of jobs, the NAs slightly increase. Under medium job variety (Figure 6.9 graphic b.), for X_{max}, X_{min}, X_{median} and $X_{average}$, the NA mean values are less than 1% and their standard deviations are less than 0.5%. For X_w and X_r, the NA mean values are around 0.6% and their standard deviations are around 1%. ***And when we increase the number of machines, the NAs are of the same order of magnitude, the changes are not notifiable.*** Under medium workshop size (**??** graphic a.), X_{max}, X_{min}, X_{median} and $X_{average}$, the mean value of new assignment are less than 0.1% and their standard deviations are less than 0.5%. For X_w and X_r, the mean values of new assignment are around 0.4% and their standard deviations are around 1%.

When we increase the degree of uncertainty, we observe a small increase of the NAs but the values are very small, the mean values are less than 2% (see e.g. Figure 6.10).

FIGURE 6.10: Number of new assignments under medium and high uncertainty degrees in the splitting case



a. (m=3,n=10,d=2)

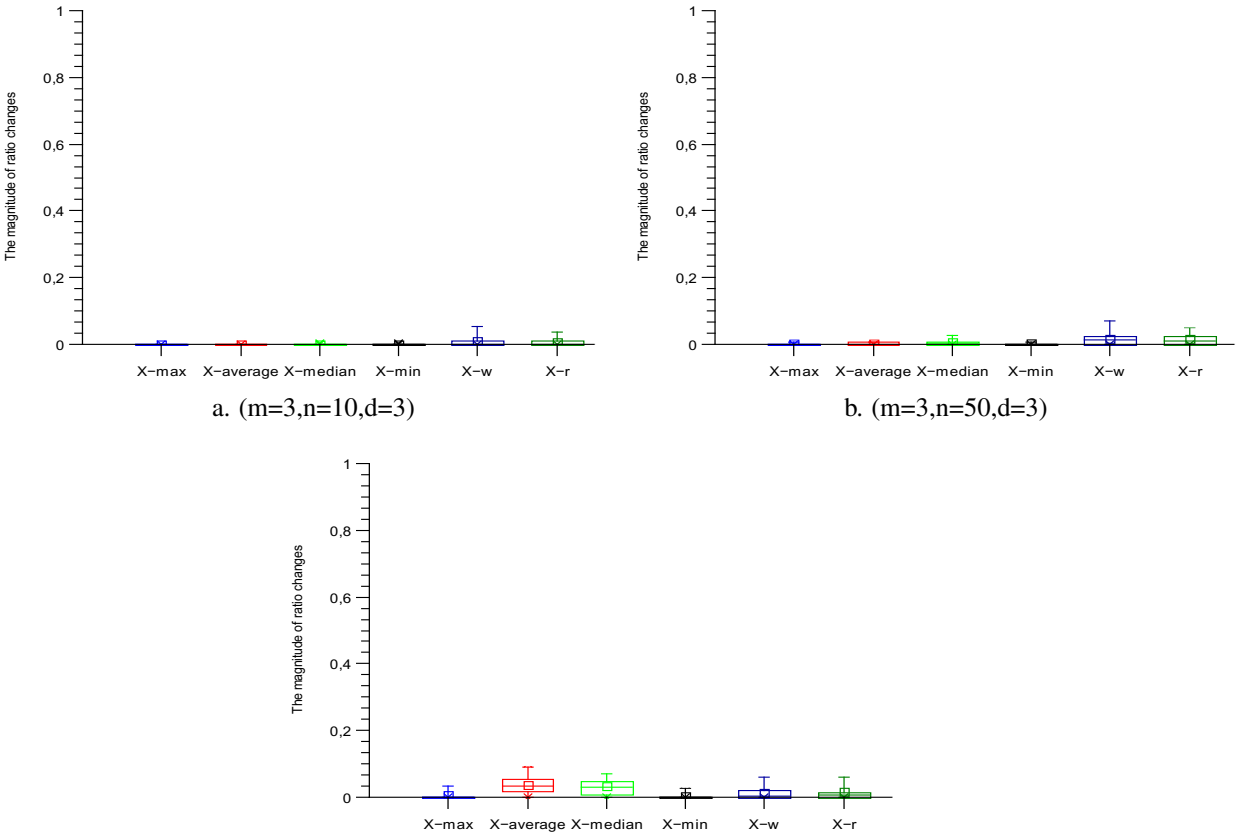
b. (m=3,n=50,d=2)

c. (m=7,n=10,d=2)

d. (m=3,n=10,d=3)

f- The magnitude of ratio changes

FIGURE 6.11: Magnitude of ratio change under low uncertainty degree in the splitting case



a. (m=3,n=10,d=3)

b. (m=3,n=50,d=3)

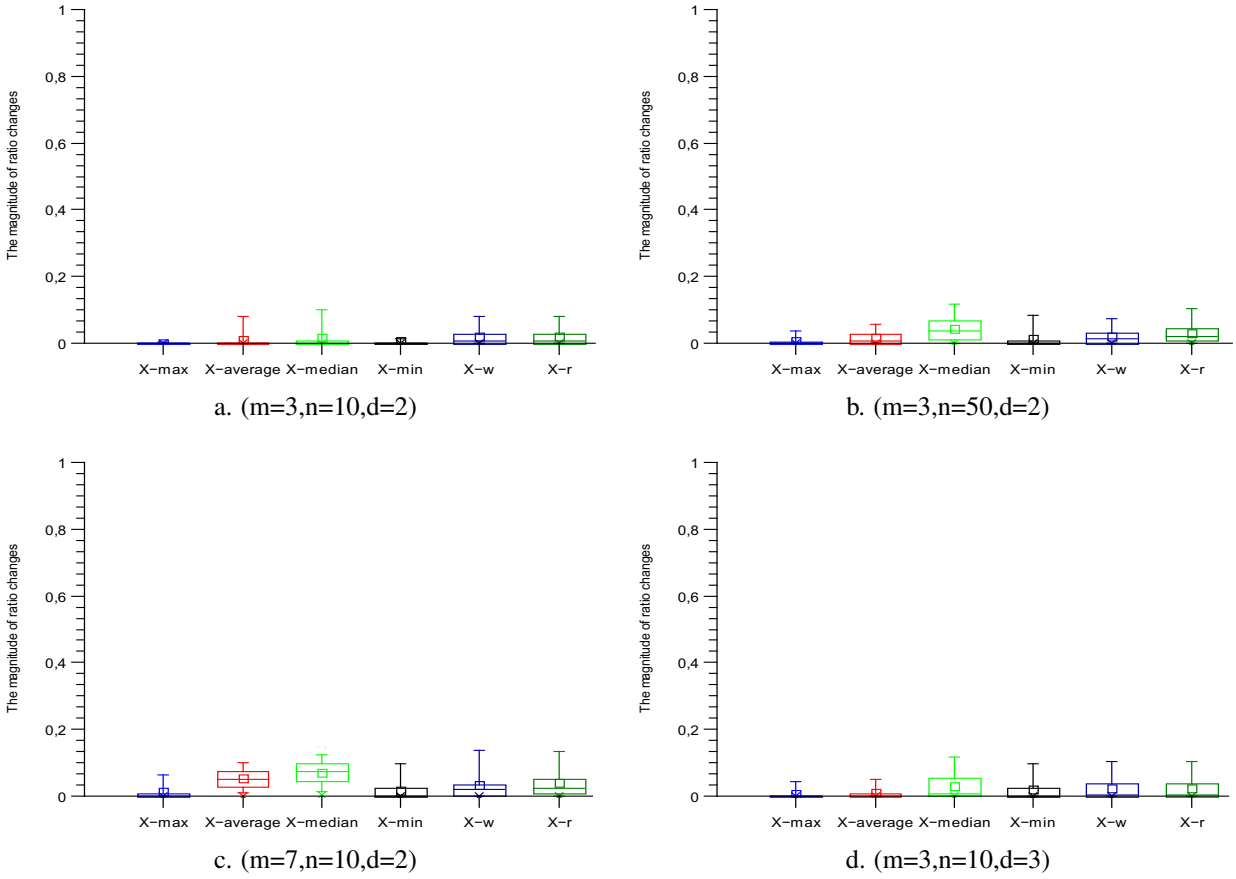
c. (m=7,n=10,d=1)

Under low uncertainty degree, all the solutions are stable according to the MC***s.*** In small size workshops under small job variety (Figure 6.11 graphic a.), for X_{max}, X_{min}, $X_{average}$ and X_{median}, the MC mean values are almost null. For X_w and X_r, the MC mean values are around 1% with a standard deviation that is equal to 1.5%.

When we increase the number of jobs, we remark that the MC ***increase is insignificant*** (*e.g.* Figure 6.11 graphic b.). ***But, when we increase the number of machines, we observe that the*** MC***s increase for*** $X_{average}$***,*** X_{median}***,*** X_w ***and*** X_r ***is relatively significant but the values are small.*** For example, in medium size workshops under small job variety (Figure 6.11 graphic c.). For X_{max} and X_{min}, the MC mean values and standard deviations are respectively equal to 0.5% and 0.3%. For X_w and X_r, the MC mean value is equal to 1.5% with standard deviations that are equal to 2%. And for $X_{average}$ and X_{median}, the average value of ratio magnitude change is equal to 3% with standard deviations that are equal to 2%.

When we increase the degree of uncertainty, the results are very similar to the case of low uncertainty degree (e.g. Figure 6.12).

FIGURE 6.12: Magnitude of ratio change under medium and high uncertainty degrees in the splitting case



a. (m=3,n=10,d=2)

b. (m=3,n=50,d=2)

c. (m=7,n=10,d=2)

d. (m=3,n=10,d=3)

6.3.3 Results synthesis and analysis

The results show that according to the performance stability indicator, we can conclude that under low uncertainty degree, all the solutions can be considered as stable: the worst-case makespan deviations are insignificant. Indeed, the scenarios have similar structure and consequently the worst case scenario is very similar to the considered scenarios.

Under medium uncertainty degree, X_{max}, X_{min}, $X_{average}$ and X_{median} are the most stable. X_w can be considered as moderately stable while X_r can become unstable especially when we increase the number of jobs. And under high uncertainty degree, X_{max}, $X_{average}$ and X_{median} are the most stable. X_{min} is quite stable. X_w and X_r are the less stable. These observations can be explained by the fact that the worst-case makespans under X_w and X_r are optimized considering the constraints of the scenario set, they take smaller values compared to the other solution makespans. In contrary, the worst-case makespans of the other solutions are already high and consequently, the deviations are less important.

According to the structure stability, we can conclude that:

- Under small uncertainty, X_{max}, X_{min} can be considered as the most stable as their number of perturbed jobs, of perturbed machines, of changed ratios and their magnitude of ratio changes are very small. X_r and

X_w can be considered as quite stable: even when the number of changed ratios is important, the magnitude of ratio changes are very small. $X_{average}$ and X_{median} are the less stable because under these solutions the numbers of perturbed jobs and perturbed machines are important, besides the number of changed ratios and the magnitude of changed ratios are significant.

The results also show that the structure stability of the solutions is highly affected by the degree of uncertainty increase and the number of machines. The solution structures become less stable. In the former case, this property is due to the fact that the new scenario under high uncertainty can be very different from the scenarios considered to compute the solution in the previous iteration, which lead to two solution structures that are very different. When we increase the number of machines, the number of alternative solutions increases and consequently the structures can be very different. The ranking of the solutions is not affected. Contrarily to the previous cases, the solution structure becomes more stable as the number of jobs increases due to the fact that the number of alternative solutions become small.

6.4 Conclusion

In this chapter, we proposed an approach to evaluate the stability of the structure and the performance of the robust solutions when adding a new scenario. In this approach, we used a 2 step-algorithm. In the first step, a set of robust solutions is computed according to the algorithms and formulations given in the previous chapters, and in the second step, additional calculations are performed in order to investigate how these solution structures and performances depend on changes in the set of potential scenarios. We evaluate the performance stability by computing the deviation of the worst-case makespan from the previous one when considering an additional scenario. And we evaluate the stability of the structure by computing different stability measures namely, the number of perturbed jobs and machines, the number of changed ratios, the number of new assignments and cancellations and the magnitude of ratio changes..

The computational results in the case of splitting show that under a new scenario, all the robust solutions are performance stable and globally they are structure stable. The worst-case makespan deviations and the magnitude of ratio changes are not significant. X_{max}, X_{min} and X_w are the most stable. $X_{average}$ and X_{median} and X_r are less stable than the former ones. The structure stability of the solutions decreases as the degree of uncertainty increase. In fact, when the scenarios become very different, the solution structures also become significantly different. Besides, as the number of machines increases, the structure stability decreases. This could be explained by the fact that the number of potential solutions is more important when the number of machines is important. Contrarily to the degree of uncertainty and the number of machine effects, the solution structures became more stable as the number of jobs increases. We would extend the computational experiments to the preemptive case.

Conclusion and Prospects

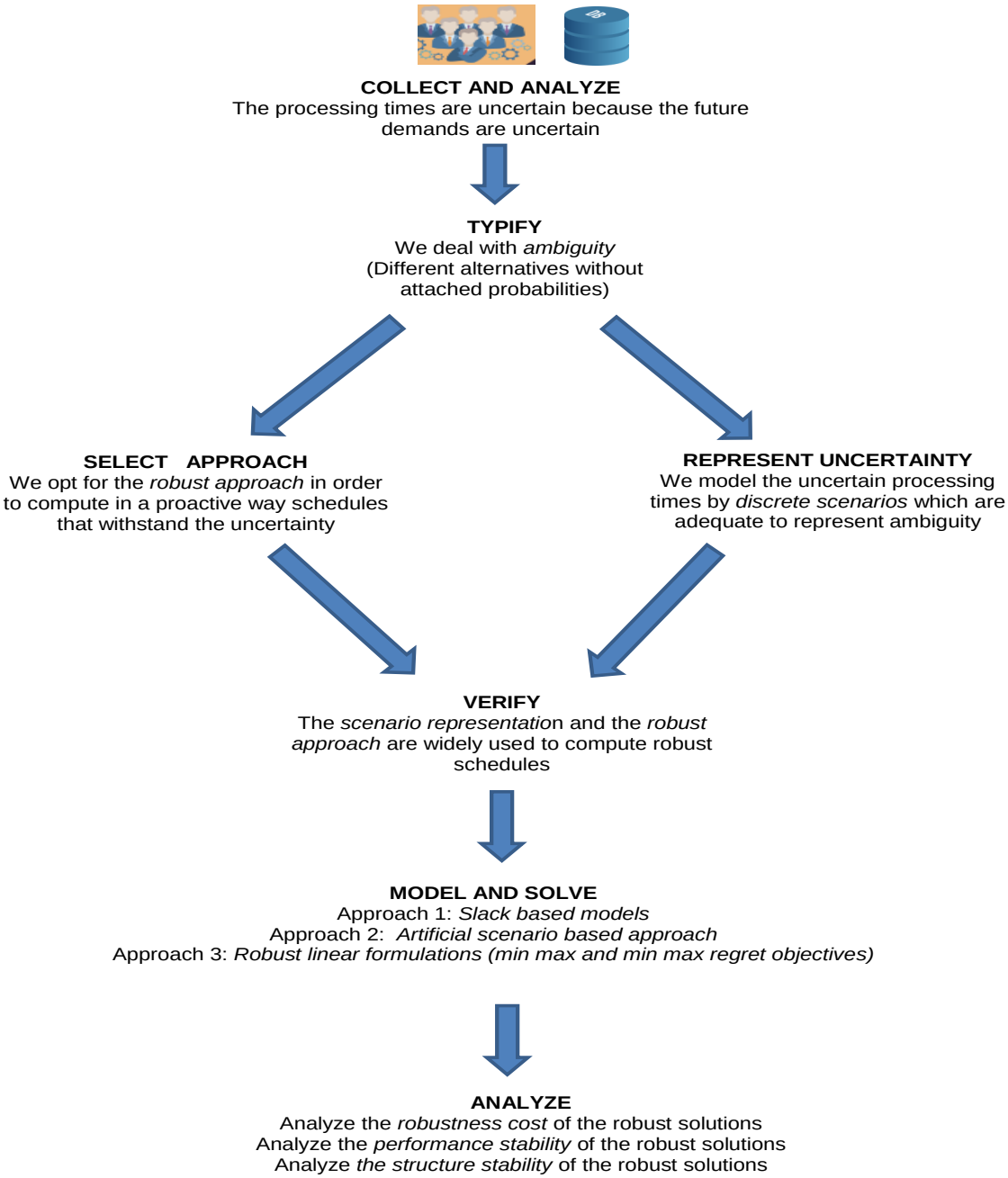
The classical scheduling algorithms compute solutions assuming a full knowledge about data before the process of scheduling starts. However, the uncertainties and disruptions are prevalent in scheduling applications. The study of robustness in scheduling allows to take into account these uncertainties and disruptions when they affect the scheduling problem. Throughout this thesis, we have been interested in scheduling on parallel machines under uncertain processing times. In the first case study, the splitting is tolerated, *i.e.*, we consider that each job can be split into continuous sub-jobs processed independently on the machines so as to finish the processing of all demands as soon as possible. In the second case study, we consider the non-overlapping splitting that is termed in literature as preemption. When the objective is to minimize the makespan, the deterministic linear formulations that assume certain processing times allow solving these problems in polynomial times. But, they suffer from limitations to hedge against the uncertain processing times: they lead to infeasible or suboptimal schedules.

The methodology and the contributions proposed in this thesis can be represented according to the general framework of optimization under uncertainty (see Chapter 1) given in Figure 6.13.

Accordingly, we considered that the uncertainty of processing times is due to uncertain demands. In fact, the decision-maker deals with different potential futures without attached probabilities. Thus, the demand uncertainty can be typified as ambiguity and consequently, the processing time uncertainty is also an ambiguity. Based on the matrix of uncertainty type representations, the discrete scenario are suitable to represent the uncertain processing times. Thus, we modelled the uncertain processing times by a finite set of potential scenarios. Furthermore, we showed through an example that there is a need for a proactive schedule that takes into account different potential scenarios instead of a schedule that is updated each time that the real scenario is revealed. For these reasons, we opted for the robust scheduling approach in order to compute, in a proactive way, solutions that withstand the uncertain processing times. The literature in Chapter 2 also showed that the robust scheduling approach under discrete scenarios is effective to compute robust schedules under uncertain parameters.

Over this thesis, we used different algorithms and models to compute robust schedules under the discrete scenario of processing times. Firstly, we showed that under the slack based approach, when we aim to enforce the feasibility of the schedule under different potential scenarios by tolerating the violations of some processing requirements, the overproduction or the underproduction are very significant (both in splitting and preemption). To avoid these consequences, we proposed a second approach called "the artificial scenario based approach" that is summed up in a general algorithm composed of 4 steps. Under this approach, we built a set of feasible solutions based on a set of artificial scenario solutions that represent special policies such as the maximal scenario, the average scenario, the median scenario, *etc.*. We applied these solutions to the

FIGURE 6.13: Application of the general framework to the parallel machines scheduling problems under uncertain processing times



potential scenarios and then computed their local and global performances. We ranked the solutions based on the classical robustness measures presented and reviewed in Chapter 2, *i.e.*, the worst-case makespan and the maximal regret.

We showed that the artificial scenario solutions insured the feasibility of all the constraints without overproduction or underproductions, and besides they can provide good performances especially the artificial scenario solution X_{max}. However, the computational results showed that the robustness costs of these solutions are not satisfying. Therefore, in Chapters 4 and 5, we proposed to compute optimal robust solutions under splitting and respectively preemption. We applied the robust discrete optimization framework developed by Kouvelis and Yu (1997) to compute the optimal robust solutions for the makespan minimization on unrelated parallel machines with job splitting (*resp.* preemption) under uncertain processing times. To insure the feasibility, we used assignment ratios as decision variables instead of the temporal variables in order to insure the feasibility of the constraints under the set of scenarios. This variable change allowed us to deal with the uncertain processing times in the objective instead of the equality constraints and to provide linear formulations to compute robust optimal solutions in polynomial times: we respectively minimized the worst-case makespan (min-max objective) and the maximal regret (min-max regret objective).

TABLE 6.4: Complexity of min max parallel scheduling problems under discrete scenario uncertainty

D(I)mM(R)$(\alpha\|\beta\|\gamma;\theta)$	*Complexities*	*References*
DmM$(Pm\|Split\|C_{max};p_j)$	Polynomial	Chapter 4
DmMR$(Pm\|Split\|C_{max};p_j)$	Polynomial	Chapter 4
DmM$(Qm\|Split\|C_{max};p_j)$	Polynomial	Chapter 4
DmMR$(Qm\|Split\|C_{max};p_j)$	Polynomial	Chapter 4
DmM$(Rm\|Split\|C_{max};p_j)$	Polynomial	Chapter 4
DmMR$(Rm\|Split\|C_{max};p_j)$	Polynomial	Chapter 4
DmM$(Rm\|pmtn\|C_{max};p_j)$	Polynomial	Chapter 5
DmMR$(Rm\|pmtn\|C_{max};p_j)$	Polynomial	Chapter 5

To analyse the results provided by these approaches, we made extensive computational tests under which we distinguished different instance sizes and different uncertainty degrees. As first analysis, we focused in the robustness cost of the robust solutions. To evaluate the robustness cost of each solution, we define an indicator that measures the relative deviation between the worst-case makespan and the nominal makespan (*i.e.* the optimal makespan under the nominal scenario). The computational results showed that, in both splitting and preemptive cases, the robustness costs of the robust solutions increase as the number of scenarios increases or the degree of uncertainty increases. Indeed, when we need to cover more potential scenarios, we need to pay additional cost to be protected against the uncertain. But, most of the robust solutions follow logarithmic trend lines which means that the robustness cost increases quickly when we cover the first potential scenario besides the nominal one (the initial robustness cost) then for each additional scenario, we pay a marginal cost.

Under small uncertainty degree, for X_{max}, X_{min}, $X_{average}$, X_{median}, X_w and X_r, the trend lines as increasing logarithms *i.e.*, each percent change in the number of scenarios is associated with a change of $0.01b$ in the robustness cost where b is the rate of increase of the logarithmic function.

The cost of robustness under small uncertainty degree can be considered as moderate. In the splitting case, X_w followed by X_r highly reduced the cost of robustness. X_{max}, X_{min}, $X_{average}$, X_{median} lead to very similar results, their robustness cost are quite superior to the optimal one. In the preemptive case, X_w and X_r also highly reduced the cost of robustness and besides X_{max} is more robust compared to the other artificial scenario solutions. However, compared to the case of splitting, the robustness cost of the robust solutions is more important in the preemptive case.

Under medium uncertainty, the robustness costs of all the solutions increase under both splitting and preemption but, the properties and the solution ranking are the same. Under high uncertainty degree, the cost of robustness highly increases for all the solutions. For X_{min} exceptionally, the cost of robustness becomes extremely huge and its trend line is no longer logarithmic but polynomial. This property is observed under both splitting and preemption. Once again, in the preemptive case, The artificial scenario solution X_{max} is more robust than the other artificial scenario solutions. It is followed by $X_{average}$ and X_{median}. And X_{min} solution is the less robust. X_w and X_r a lead to the lowest robustness cost.

As second analysis, we considered a new issue in robust scheduling which is the stability of the robust solution performances under a new scenario. Indeed, the discrete scenario representation that permits to compute the robust solutions is used primarily to limit the set of possible futures but, in reality the workshop will often face new scenarios. Thus, focusing on the robustness of the performance under a specific set of scenarios as a unique objective can be restrictive. In fact, the solution robustness is guaranteed only for processing time realizations that belong to the represented scenario set. Consequently, from one hand, the robust solution might have no robustness guarantee if the real scenario is a new one. On the other hand, the robust approach is not useful to help the decision-maker understand how much the objective could deviate, nor how much adjustment should be taken to cover it. Indeed, depending on whether a plausible scenario is taken into account or not, the structure of the robust solution could be very different. This aspect of robustness dependency to the chosen scenarios was already pointed out by Roy (2010) who claims that "adding or removing a scenario from the set of scenarios may lead to defining very different solutions as robust". From practitioners viewpoint, the quantification of these deviations under new scenarios both in terms of structure and performance is a very important question. Moreover, schedulers confronted with data uncertainty agree that it is difficult to list all the potential scenarios. Therefore, it is crucial to identify robust solutions whose structure and performance do not deviate significantly under new scenarios or to be able to choose the desired trade-off. In other words, it is important to choose among the robust solutions those with the most stable structure and the most stable performance.

The stability of the performance under a new scenario is measured through the worst-case makespan deviation, and the structure stability is measured according to different measures: the number of perturbed jobs, the number of perturbed machines, the number of changing ratios, the number of new assignments, the number of cancelled assignments and the magnitude of ratio changes.

Overall, the computational results show that X_{max} is quite robust, performance stable and structure

stable under all uncertainty degrees. Under small and medium uncertainty degrees, X_{min} is quite robust, performance stable and structure stable but under high uncertainty degree, it is performance stable and structure stable but it is not robust. The most robust solution X_w is moderately stable as its performance stability and its structure stability decrease as the degree of uncertainty increase. X_r is also robust but less stable than X_w. In which concerns $X_{average}$ and X_{median}, these solutions are quite robust and quite stable under all the uncertainty degrees.

The results also show that the structure stability of the solutions is highly affected by the degree of uncertainty and the number of machines. The solution structures become less stable but, the ranking of the solutions is not affected. Contrarily to the previous cases, the solution structures become more stable when the number of jobs increases.

Research prospects

Following the results described in this thesis, the future works can be oriented towards the following developments.

As short term developments, we aim to address the robust preemptive problem with unique sequence. Indeed, in some decision environments, the decision-maker does not admit having different sequences (one by scenario) but, in some sectors it might be necessary to compute a robust schedule with a unique assignment and a unique sequence. The main questions that arise from this are related to the complexity of the robust integrated problem resolution. Besides, it would be also interesting to extend the approaches developed in this thesis to the non-preemptive case which is known to be NP-hard even under certainty. The evaluation of the robustness cost in this case and if it also follows logarithmic trend line would be a good research question.

As mid and long-term developments, we can consider different future works. Indeed, in a real scheduling environment, there are possible scenarios in which activities may be added or removed (new jobs or cancelled jobs), or temporal constraints may be revised. Due to the practical importance, it is therefore desirable to extend our methodologies to modelling and solving these scheduling problems under job disruptions or resource disruptions.

Furthermore, it would be interesting to integrate reactive procedures with the robust scheduling approaches. In fact, in this thesis, we solved the scheduling problems from the perspective of proactive scheduling and obtained robust execution strategy off-line. One of our research efforts aims to integrate robust models with reactive scheduling algorithms, implement the obtained execution strategy and evaluate it.

And lastly, many additional opportunities for future research can be identified when adapting our approaches to an industrial context.

Bibliography

Abumaizar, R. J. and **J. A. Svestka** (1997). Rescheduling job shops under random disruptions. *International Journal of Production Research*, **35**(7), 2065–2082.

Ahmadi, E., **M. Zandieh**, **M. Farrokh**, and **S. M. Emami** (2016). A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms. *Computers & Operations Research*, **73**, 56–66.

Ahmed, S. and **N. V. Sahinidis** (1998). Robust process planning under uncertainty. *Industrial & Engineering Chemistry Research*, **37**(5), 1883–1892.

Aissi, H., **M. A. Aloulou**, and **M. Y. Kovalyov** (2011). Minimizing the number of late jobs on a single machine under due date uncertainty. *Journal of Scheduling*, **14**(4), 351–360.

Al-Hinai, N. and **T. ElMekkawy** (2011). Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm. *International Journal of Production Economics*, **132**(2), 279–291.

Aloulou, M. A. and **F. Della Croce** (2008). Complexity of single machine scheduling problems under scenario-based uncertainty. *Operations Research Letters*, **36**(3), 338–342.

Armbruster, B. and **E. Delage** (2015). Decision making under uncertainty when preference information is incomplete. *Management Science*, **61**(1), 111–128.

Artigues, C., **R. Leus**, and **F. T. Nobibon** (2013). Robust optimization for resource-constrained project scheduling with uncertain activity durations. *Flexible Services and Manufacturing Journal*, **25**(1-2), 175–205.

Aubry, A., **M. Jacomino**, **A. Rossi**, and **M.-L. Espinouse** (2012). Maximizing the configuration robustness for parallel multi-purpose machines under setup cost constraints. *Journal of Scheduling*, **15**(4), 457– 471.

Averbakh, I. (2006). The minmax regret permutation flow-shop problem with two jobs. *European Journal of Operational Research*, **169**(3), 761–766.

Aytug, H., **M. A. Lawley**, **K. McKay**, **S. Mohan**, and **R. Uzsoy** (2005*a*). Executing production schedules in the face of uncertainties: A review and some future directions. *European Journal of Operational Research*, **161**(1), 86–110.

Aytug, H., **M. A. Lawley**, **K. McKay**, **S. Mohan**, and **R. Uzsoy** (2005*b*). Executing production schedules in the face of uncertainties: A review and some future directions. *European Journal of Operational Research*, **161**(1), 86 – 110.

Ayyub, B. M. and **G. J. Klir**, *Uncertainty modeling and analysis in engineering and the sciences*. CRC Press, 2006.

Azizoglu, M. and **O. Alagöz** (2005). Parallel-machine rescheduling with machine disruptions. *IIE Transactions*, **37**(12), 1113–1118.

Ballestin, F. and **R. Leus** (2009). Resource-constrained project scheduling for timely project completion with stochastic activity durations. *Production and Operations Management*, **18**(4), 459–474.

Bandi, C. and **D. Bertsimas** (2012). Tractable stochastic analysis in high dimensions via robust optimization. *Mathematical programming*, 1–48.

Beck, J. and **A. Davenport** (2002). A survey of techniques for scheduling with uncertainty.

Bedford, T. and **R. Cooke** (2001). Different types of uncertainty. *Probabilistic Risk Analysis: Foundations and Methods*, 17–38.

Bellman, R. E. and **L. A. Zadeh** (1970). Decision-making in a fuzzy environment. *Management science*, **17**(4), B–141.

Ben-Tal, A., **L. El Ghaoui**, and **A. Nemirovski**, *Robust optimization*. Princeton Series in Applied Mathematics. Princeton University Press, 2009.

Ben-Tal, A. and **A. Nemirovski** (1998). Robust convex optimization. *Mathematics of Operations Research*, **23**(4), 769–805.

Ben-Tal, A. and **A. Nemirovski** (2000). Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical programming*, **88**(3), 411–424.

Bentaha, M. L., **A. Dolgui**, and **O. Battaïa** (2015). A bibliographic review of production line design and balancing under uncertainty. *IFAC-PapersOnLine*, **48**(3), 70–75.

Bertsimas, D. and **D. B. Brown** (2009). Constructing uncertainty sets for robust linear optimization. *Operations research*, **57**(6), 1483–1495.

Bertsimas, D., **D. B. Brown**, and **C. Caramanis** (2011). Theory and applications of robust optimization. *SIAM review*, **53**(3), 464–501.

Bertsimas, D., **D. A. Iancu**, and **P. A. Parrilo** (2010). Optimality of affine policies in multistage robust optimization. *Mathematics of Operations Research*, **35**(2), 363–394.

Bertsimas, D. and **M. Sim** (2003). Robust discrete optimization and network flows. *Mathematical programming*, **98**(1-3), 49–71.

Bertsimas, D. and **M. Sim** (2004). The price of robustness. *Operations research*, **52**(1), 35–53.

Billaut, J.-C., **A. Moukrim**, and **E. Sanlaville**, *Flexibility and Robustness in Scheduling*. Wiley Online Library, 2008.

Bilyk, A. and **L. Mönch** (2012). A variable neighborhood search approach for planning and scheduling of jobs on unrelated parallel machines. *Journal of Intelligent Manufacturing*, **23**(5), 1621–1635.

Birge, J. R. and **F. Louveaux**, *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

Błażewicz, J., **K. H. Ecker**, **G. Schmidt**, and **J. Weglarz**, *Scheduling in computer and manufacturing systems*. Springer Science & Business Media, 2012.

Charnes, A. and **W. W. Cooper** (1959). Chance-constrained programming. *Management science*, **6**(1), 73–79.

Cheng, T. and **C. Sin** (1990). A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research*, **47**(3), 271–292.

Chiang, T.-C., **H.-C. Cheng**, and **L.-C. Fu** (2010). A memetic algorithm for minimizing total weighted tardiness on parallel batch machines with incompatible job families and dynamic job arrival. *Computers & Operations Research*, **37**(12), 2257–2269.

Colyvan, M. (2008). Is probability the only coherent approach to uncertainty? *Risk Analysis*, **28**(3), 645–652.

Conde, E. (2010). A 2-approximation for minmax regret problems via a mid-point scenario optimal solution. *Operations Research Letters*, **38**(4), 326–327.

Conde, E. (2014). A mip formulation for the minmax regret total completion time in scheduling with unrelated parallel machines. *Optimization Letters*, **8**(4), 1577–1589.

Courtney, H. (2003). Decision-driven scenarios for assessing four levels of uncertainty. *Strategy & Leadership*, **31**(1), 14–22.

Cowling, P. and **M. Johansson** (2002). Using real time information for effective dynamic scheduling. *European Journal of Operational Research*, **139**(2), 230–244.

Curry, J. and **B. Peters** (2005). Rescheduling parallel machines with stepwise increasing tardiness and machine assignment stability objectives. *International Journal of Production Research*, **43**(15), 3231–3246.

Cwik, M. and **J. Józefczyk** (2015). Evolutionary algorithm for minmax regret flow-shop problem. *Management and Production Engineering Review*, **6**(3), 3.

Daniels, R. L. and **J. E. Carrillo** (1997). β-robust scheduling for single-machine systems with uncertain processing times. *IIE transactions*, **29**(11), 977–985.

Daniels, R. L. and **P. Kouvelis** (1995*a*). Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science*, **41**(2), 363–376.

Daniels, R. L. and **P. Kouvelis** (1995*b*). Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science*, **41**(2), 363–376.

Dantzig, G. B. *et al.* (1955). Linear programming under uncertainty. *Management Science*, **1**(3-4), 197–206.

Daskin, M. S., **S. M. Hesse**, and **C. S. Revelle** (1997). α-reliable p-minimax regret: A new model for strategic facility location modeling. *Location Science*, **5**(4), 227–246.

Davenport, A., **C. Gefflot**, and **C. Beck**, Slack-based techniques for robust schedules. *In Sixth European conference on planning*. 2014.

Davidson, K. M. (1991). Why acquisitions may not be the best route to innovation. *Journal of Business Strategy*, **12**(3), 50–52.

Davidson, P. (1996). Some misunderstanding on uncertainty in modern classical economics. *Schmidt C.(éd.), Uncertainty in Economic Thought, Edward Elgar*, 21–37.

de Farias, I. R., **H. Zhao**, and **M. Zhao** (2010). A family of inequalities valid for the robust single machine scheduling polyhedron. *Computers & Operations Research*, **37**(9), 1610–1614.

De Finetti, B., La prévision: ses lois logiques, ses sources subjectives. *In Annales de l'institut Henri Poincaré*, volume 7. 1937.

Dempster, A. P. (1967). Upper and lower probabilities induced by a multivalued mapping. *The annals of mathematical statistics*, 325–339.

Dequech, D. (2000). Fundamental uncertainty and ambiguity. *Eastern Economic Journal*, **26**(1), 41–60.

Dequech, D. (2011). Uncertainty: a typology and refinements of existing concepts. *Journal of economic issues*, **45**(3), 621–640.

Der Kiureghian, A. and **O. Ditlevsen** (2009). Aleatory or epistemic? does it matter? *Structural Safety*, **31**(2), 105–112.

Dolgui, A. and **S. Kovalev** (2012). Scenario based robust line balancing: Computational complexity. *Discrete Applied Mathematics*, **160**(13-14), 1955–1963.

Dong, Y.-H. and **J. Jang** (2012). Production rescheduling for machine breakdown at a job shop. *International Journal of Production Research*, **50**(10), 2681–2691.

Dosi, G. and **M. Egidi** (1991). Substantive and procedural uncertainty. *Journal of Evolutionary Economics*, **1**(2), 145–168.

Dow, S. (2016). Uncertainty: A diagrammatic treatment. *Economics*, **10**(3), 1.

Drwal, M. (2017). Min-max regret scheduling to minimize the total weight of late jobs with interval uncertainty. *arXiv preprint arXiv:1706.03103*.

Drwal, M. and **R. Rischke** (2016). Complexity of interval minmax regret scheduling on parallel identical machines with total completion time criterion. *Operations Research Letters*, **44**(3), 354–358.

Dubois, D., **H. Fargier**, and **P. Fortemps** (2003). Fuzzy scheduling: Modelling flexible constraints vs. coping with incomplete knowledge. *European Journal of Operational Research*, **147**(2), 231–252.

Dubois, D. and **H. Prade** (2009). Formal representations of uncertainty. *Decision-Making Process: Concepts and Methods*, 85–156.

Ebrahimi, M., **S. F. Ghomi**, and **B. Karimi** (2014). Hybrid flow shop scheduling with sequence dependent family setup time and uncertain due dates. *Applied Mathematical Modelling*, **38**(9), 2490–2504.

El Ghaoui, L., **F. Oustry**, and **H. Lebret** (1998). Robust solutions to uncertain semidefinite programs. *SIAM Journal on Optimization*, **9**(1), 33–52.

Ellsberg, D. (1961). Risk, ambiguity, and the savage axioms. *The quarterly journal of economics*, 643–669.

Ferson, S. and **L. R. Ginzburg** (1996). Different methods are needed to propagate ignorance and variability. *Reliability Engineering & System Safety*, **54**(2), 133–144.

Freeman, C. (1982). The economics of industrial innovation. *University of Illinois at Urbana-Champaign's Academy for Entrepreneurial Leadership Historical Research Reference in Entrepreneurship*.

Gao, H., *Building robust schedules using temporal protection: an empirical study of constraint based scheduling under machine failure uncertainty.*. University of Toronto, 1996.

Georghiou, A., **W. Wiesemann**, and **D. Kuhn** (2015). Generalized decision rule approximations for stochastic programming via liftings. *Mathematical Programming*, **152**(1-2), 301–338.

Gonzalez, T. (1979). A note on open shop preemptive schedules. *IEEE Transactions on Computers*, (10), 782–786.

Gonzalez, T. and **S. Sahni** (1976). Open shop scheduling to minimize finish time. *Journal of the ACM (JACM)*, **23**(4), 665–679.

Goren, S. and **I. Sabuncuoglu** (2008). Robustness and stability measures for scheduling: single-machine environment. *IIE Transactions*, **40**(1), 66–83.

Goren, S. and **I. Sabuncuoglu** (2009). Optimization of schedule robustness and stability under random machine breakdowns and processing time variability. *IIE Transactions*, **42**(3), 203–220.

Gorissen, B. L., **İ. Yanıkoğlu**, and **D. den Hertog** (2015). A practical guide to robust optimization. *Omega*, **53**, 124–137.

Graham, R. L., **E. L. Lawler**, **J. K. Lenstra**, and **A. R. Kan** (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of discrete mathematics*, **5**, 287–326.

Guinand, F., **A. Moukrim**, and **E. Sanlaville** (2004). Sensitivity analysis of tree scheduling on two machines with communication delays. *Parallel Computing*, **30**(1), 103–120.

Gupta, S. K. and **J. Rosenhead** (1968). Robustness in sequential investment decisions. *Management science*, **15**(2), B–18.

Halpern, J. Y., *Reasoning about uncertainty*. MIT press, 2005.

Hammonds, J., **F. Hoffman**, and **S. Bartell** (1994). An introductory guide to uncertainty analysis in environmental and health risk assessment. *Technical Rep. No. ES/ER/TM-35*, **1**.

Hancerliogullari, G., **G. Rabadi**, **A. H. Al-Salem**, and **M. Kharbeche** (2013). Greedy algorithms and metaheuristics for a multiple runway combined arrival-departure aircraft sequencing problem. *Journal of Air Transport Management*, **32**, 39–48.

He, W. and **D.-h. Sun** (2013). Scheduling flexible job shop problem subject to machine breakdown with route changing and right-shift strategies. *The International Journal of Advanced Manufacturing Technology*, 1–14.

Helton, J. C. (1994). Treatment of uncertainty in performance assessments for complex systems. *Risk analysis*, **14**(4), 483–511.

Herroelen, W. and **R. Leus** (2004*a*). The construction of stable project baseline schedules. *European Journal of Operational Research*, **156**(3), 550–565.

Herroelen, W. and **R. Leus** (2004*b*). Robust and reactive project scheduling: a review and classification of procedures. *International Journal of Production Research*, **42**(8), 1599–1620.

Herroelen, W. and **R. Leus** (2005). Project scheduling under uncertainty: Survey and research potentials. *European journal of operational research*, **165**(2), 289–306.

Hora, S. C. (1996). Aleatory and epistemic uncertainty in probability elicitation with an example from hazardous waste management. *Reliability Engineering & System Safety*, **54**(2), 217–223.

Hosseini, N. and **R. Tavakkoli-Moghaddam** (2013). 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*, 1–16.

Iancu, D. A. and **N. Trichakis** (2013). Pareto efficiency in robust optimization. *Management Science*, **60**(1), 130–147.

Jain, A. and **H. Elmaraghy** (1997). Production scheduling/rescheduling in flexible manufacturing. *International Journal of Production Research*, **35**(1), 281–309.

Ji, Q., **Y. Wang**, and **X. Hu** (2016). Optimal production planning for assembly systems with uncertain capacities and random demand. *European Journal of Operational Research*, **253**(2), 383–391.

Jorge Leon, V., **S. David Wu**, and **R. H. Storer** (1994). Robustness measures and robust scheduling for job shops. *IIE transactions*, **26**(5), 32–43.

Kacprzyk, J. and **S. A. Orlovski**, *Optimization models using fuzzy sets and possibility theory*, volume 4. Springer Science & Business Media, 2013.

Kalaï, R., **M. A. Aloulou**, **P. Vallin**, and **D. Vanderpooten** (2010). Lexicographic α-robustness: an application to the 1-median problem. *RAIRO-Operations Research*, **44**(2), 119–138.

Kalaï, R. and **C. Lamboray** (2007). l'α-robustesse lexicographique: une relaxation de la β-robustesse.

Kali, P., *Stochastic programming*. Springer, .

Kall, P., Chance constrained programming. *In Stochastic Linear Programming*. Springer, 1976, 79–92.

Kaplan, S. and **G. Rabadi** (2012). Exact and heuristic algorithms for the aerial refueling parallel machine scheduling problem with due date-to-deadline window and ready times. *Computers & Industrial Engineering*, **62**(1), 276–285.

Kaplan, S. and **G. Rabadi** (2013). Simulated annealing and metaheuristic for randomized priority search algorithms for the aerial refuelling parallel machine scheduling problem with due date-to-deadline windows and release times. *Engineering Optimization*, **45**(1), 67–87.

Kaplan, S. and **G. Rabadi** (2015). Minimising the total weighted tardiness and instability for the parallel machine re-scheduling problem with deadlines and ready times. *International Journal of Planning and Scheduling*, **2**(2), 87–109.

Karmarkar, N., A new polynomial-time algorithm for linear programming. *In Proceedings of the sixteenth annual ACM symposium on Theory of computing*. ACM, 1984.

Kasperski, A. (2005). Minimizing maximal regret in the single machine sequencing problem with maximum lateness criterion. *Operations Research Letters*, **33**(4), 431–436.

Kasperski, A., **A. Kurpisz**, and **P. Zieliński** (2012). Approximating a two-machine flow shop scheduling under discrete scenario uncertainty. *European Journal of Operational Research*, **217**(1), 36–43.

Kasperski, A. and **P. Zielienski** (2008). A 2-approximation algorithm for interval data minmax regret sequencing problems with the total flow time criterion. *Operations Research Letters*, **36**(3), 343–344.

Kasperski, A. and **P. Zieliński**, Sequencing problems with uncertain parameters and the owa criterion. *In Operations Research Proceedings 2013*. Springer, 2014, 223–229.

Kasperski, A. and **P. Zieliński** (2016). Single machine scheduling problems with uncertain parameters and the owa criterion. *Journal of Scheduling*, **19**(2), 177–190.

Katragjini, K., **E. Vallada**, and **R. Ruiz** (2013). Flow shop rescheduling under different types of disruption. *International Journal of Production Research*, **51**(3), 780–797.

Keynes, J. M. (1936). The general theory of interest, employment and money.

Khachian, L. G. (1979). A polynomial algorithm in linear programming. *Doklady Akademii Nauks SSR*, **244**(5), 1093–1096.

Klir, G. and **M. Wierman** (1998). Uncertainty-based information: Variables of generalized information theory. *Physica-Verlag, New York, NY*.

Knight, F. H. (1921). Risk, uncertainty andprofit. *New York: AM Kelley*, **1964**.

Kochenderfer, M. J., **C. Amato**, **G. Chowdhary**, **J. P. How**, **H. J. D. Reynolds**, **J. R. Thornton**, **P. A. Torres-Carrasquillo**, **N. K. Üre**, and **J. Vian**, *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2015.

Kolmogorov, A. N. (1933). Foundations of probability.

Koopmans, T. C. (1957). Allocation of resources and the price system.

Kouvelis, P., **R. L. Daniels**, and **G. Vairaktarakis** (2000). Robust scheduling of a two-machine flow shop with uncertain processing times. *IIE Transactions*, **32**(5), 421–432.

Kouvelis, P. and **G. Yu**, *Robust discrete optimization and its applications*, volume 14. Springer Science & Business Media, 1997.

Kuhn, D., **W. Wiesemann**, and **A. Georghiou** (2011). Primal and dual linear decision rules in stochastic and robust optimization. *Mathematical Programming*, **130**(1), 177–209.

Kwakkel, J. H., **W. E. Walker**, and **V. Marchau** (2010). Adaptive airport strategic planning. *European Journal of Transport and Infrastructure Research (EJTIR), 10 (3), 2010.*

Lai, T. and **Y. N. Sotskov** (1999). Sequencing with uncertain numerical data for makespan minimisation. *Journal of the Operational Research Society*, **50**(3), 230–243.

Lawler, E. L. and **J. Labetoulle** (1978). On preemptive scheduling of unrelated parallel processors by linear programming. *Journal of the ACM (JACM)*, **25**(4), 612–619.

Lebedev, V. and **I. Averbakh** (2006). Complexity of minimizing the total flow time with interval data and minmax regret criterion. *Discrete Applied Mathematics*, **154**(15), 2167–2177.

Lenstra, J. K. and **A. Rinnooy Kan** (1978). Complexity of scheduling under precedence constraints. *Operations Research*, **26**(1), 22–35.

Leus, R. and **W. Herroelen** (2004). Stability and resource allocation in project planning. *IIE transactions*, **36**(7), 667–682.

Leus, R. and **W. Herroelen** (2005). The complexity of machine scheduling for stability with a single disrupted job. *Operations Research Letters*, **33**(2), 151–156.

Li, Z. and **C. A. Floudas** (2014). A comparative theoretical and computational study on robust counterpart optimization: Iii. improving the quality of robust solutions. *Industrial & engineering chemistry research*, **53**(33), 13112–13124.

Liao, Z. and **J. Rittscher** (2007). A multi-objective supplier selection model under stochastic demand conditions. *International Journal of Production Economics*, **105**(1), 150–159.

Lin, X., **S. L. Janak**, and **C. A. Floudas** (2004). A new robust optimization approach for scheduling under uncertainty:: I. bounded uncertainty. *Computers & chemical engineering*, **28**(6), 1069–1085.

Liu, B., *Uncertainty theory*, volume 24. Springer, 2015.

Lodwick, W. A. and **E. A. Untiedt**, *Introduction to Fuzzy and Possibilistic Optimization.*. Springer, 2010.

Luce, R. D. and **H. Raiffa**, *Games and Decisions: Introduction and Critical Survey*. Courier Corporation, 1957.

Makridakis, S. and **N. Taleb** (2009). Decision making and planning under low levels of predictability. *International Journal of Forecasting*, **25**(4), 716–733.

Malve, S. and **R. Uzsoy** (2007). A genetic algorithm for minimizing maximum lateness on parallel identical batch processing machines with dynamic job arrivals and incompatible job families. *Computers & Operations Research*, **34**(10), 3016–3028.

Mastrolilli, M., **N. Mutsanas**, and **O. Svensson** (2013). Single machine scheduling with scenarios. *Theoretical Computer Science*, **477**, 57–66.

McKay, K. N. and **V. C. Wiers** (1999). Unifying the theory and practice of production scheduling. *Journal of Manufacturing Systems*, **18**(4), 241–255.

Mehta, S. V. (1999). Predictable scheduling of a single machine subject to breakdowns. *International Journal of Computer Integrated Manufacturing*, **12**(1), 15–38.

Mehta, S. V. and **R. M. Uzsoy** (1998). Predictable scheduling of a job shop subject to breakdowns. *Robotics and Automation, IEEE Transactions on*, **14**(3), 365–378.

Mokotoff, E. (2001). Parallel machine scheduling problems: A survey. *Asia-Pacific Journal of Operational Research*, **18**(2), 193.

Morgan, M. G. and **M. Henrion** (1990). Uncertainty: a guide to dealing with uncertainty in quantitative risk and policy analysis cambridge university press. *New York, New York, USA*.

Moukrim, A., **E. Sanlaville**, and **F. Guinand** (2003). Parallel machine scheduling with uncertain communication delays. *RAIRO-Operations Research*, **37**(1), 1–16.

Mulvey, J. M., **R. J. Vanderbei**, and **S. A. Zenios** (1995). Robust optimization of large-scale systems. *Operations research*, **43**(2), 264–281.

Nash, J. F. *et al.* (1950). Equilibrium points in n-person games.

Nemirovski, A., **A. Juditsky**, **G. Lan**, and **A. Shapiro** (2009). Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, **19**(4), 1574–1609.

Nowak, M. and **A. Tomasgard** (2007). Scenario generation and forecasting. Technical report, Working paper, Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway.

Oberkampf, W. L., **T. G. Trucano**, and **C. Hirsch** (2004). Verification, validation, and predictive capability in computational engineering and physics. *Applied Mechanics Reviews*, **57**(5), 345–384.

Ouelhadj, D. and **S. Petrovic** (2009). A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling*, **12**(4), 417–431.

Özlen, M. and **M. Azizoğlu** (2009). Generating all efficient solutions of a rescheduling problem on unrelated parallel machines. *International Journal of Production Research*, **47**(19), 5245–5270.

Ozlen, M. and **M. Azizoğlu** (2011). Rescheduling unrelated parallel machines with total flow time and total disruption cost criteria. *Journal of the Operational Research Society*, **62**(1), 152–164.

Pei, J., **X. Liu**, **W. Fan**, **P. M. Pardalos**, **A. Migdalas**, and **S. Yang** (2016). Scheduling jobs on a single serial-batching machine with dynamic job arrivals and multiple job types. *Annals of Mathematics and Artificial Intelligence*, **76**(1-2), 215–228.

Pereira, J. (2016). The robust (minmax regret) single machine scheduling with interval processing times and total weighted completion time objective. *Computers & Operations Research*, **66**, 141–152.

Petrovic, D., **R. Roy**, and **R. Petrovic** (1999). Supply chain modelling using fuzzy sets. *International journal of production economics*, **59**(1), 443–453.

Petrovic, S., **C. Fayad**, **D. Petrovic**, **E. Burke**, and **G. Kendall** (2008). Fuzzy job shop scheduling with lot-sizing. *Annals of Operations Research*, **159**(1), 275–292.

Pinedo, M. L., *Scheduling: theory, algorithms, and systems*. Springer-Verlag New York, 2012.

Potts, C. N. and **L. Van Wassenhove** (1992). Integrating scheduling with batching and lot-sizing: a review of algorithms and complexity. *Journal of the Operational Research Society*, **43**, 395 – 406.

Rahmani, D. and **M. Heydari** (2014). Robust and stable flow shop scheduling with unexpected arrivals of new jobs and uncertain processing times. *Journal of Manufacturing Systems*, **33**(1), 84–92.

Ramsey, F. P. (1931). Truth and probability (1926). *The foundations of mathematics and other logical essays*, 156–198.

Rao, K. and **G. R. Janardhana** (2016). Rescheduling activities in face of disruption in house hold goods manufacturing supply chain. *International Journal of Applied Industrial Engineering (IJAIE)*, **3**(2), 47–65.

Rockafellar, R. T. and **R. J.-B. Wets** (1991). Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of operations research*, **16**(1), 119–147.

Rosenblatt, M. J. and **H. L. Lee** (1987). A robustness approach to facilities design. *International Journal of Production Research*, **25**(4), 479–486.

Rosenhead, J., **M. Elton**, and **S. K. Gupta** (1972*a*). Robustness and optimality as criteria for strategic decisions. *Journal of the Operational Research Society*, **23**(4), 413–431.

Rosenhead, J., **M. Elton**, and **S. K. Gupta** (1972*b*). Robustness and optimality as criteria for strategic decisions. *Operational Research Quarterly*, 413–431.

Rossi, A. (2003). *Ordonnancement en milieu incertain, mise en oeuvre d'une démarche robuste*. Ph.D. thesis, Institut National Polytechnique de Grenoble-INPG.

Rossi, A. (2010). A robustness measure of the configuration of multi-purpose machines. *International journal of production research*, **48**(4), 1013–1033.

Rowe, W. D. (1994). Understanding uncertainty. *Risk analysis*, **14**(5), 743–750.

Roy, B. (2010). Robustness in operational research and decision aiding: A multi-faceted issue. *European Journal of Operational Research*, **200**(3), 629–638.

Ruszczynski, A. P. and **A. Shapiro**, *Stochastic programming*, volume 10. Elsevier Amsterdam, 2003.

Sabuncuoglu, I. and **S. Goren** (2009). Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research. *International Journal of Computer Integrated Manufacturing*, **22**(2), 138–157.

Saghafian, S. and **B. Tomlin** (2016). The newsvendor under demand ambiguity: Combining data with moment and tail information. *Operations Research*, **64**(1), 167–185.

Sahinidis, N. V. (2004). Optimization under uncertainty: state-of-the-art and opportunities. *Computers & Chemical Engineering*, **28**(6), 971–983.

Savage, L. J. (1954). The foundations of statistics.

Schütz, P., **A. Tomasgard**, and **S. Ahmed** (2009). Supply chain design under uncertainty using sample average approximation and dual decomposition. *European Journal of Operational Research*, **199**(2), 409–419.

Shackle, G. L. S., **C. F. Carter**, **J. Ford**, *et al.* (1972). Uncertainty and expectations in economics: essays in honour of gls shackle.

Shafer, G. *et al.*, *A mathematical theory of evidence*, volume 1. Princeton university press Princeton, 1976.

Shi, X., **H. Shen**, **T. Wu**, and **T. Cheng** (2014). Production planning and pricing policy in a make-to-stock system with uncertain demand subject to machine breakdowns. *European Journal of Operational Research*, **238**(1), 122–129.

Siepak, M. and **J. Józefczyk** (2014). Solution algorithms for unrelated machines minmax regret scheduling problem with interval processing times and the total flow time criterion. *Annals of Operations Research*, **222**(1), 517–533.

Silva, C. and **J. M. Magalhaes** (2006). Heuristic lot size scheduling on unrelated parallel machines with applications in the textile industry. *Computers and Industrial Engineering*, **50**(1), 76–89.

Smithson, M. (1989). Ignorance and uncertainty: Emerging paradigms.

Smithson, M. (1990). Ignorance and disasters. *International journal of mass emergencies and disasters*, **8**(3), 207–235.

Smithson, M., *Ignorance and uncertainty: emerging paradigms*. Springer Science & Business Media, 2012.

Sotskov, Y., **N. Y. Sotskova**, and **F. Werner** (1997). Stability of an optimal schedule in a job shop. *Omega*, **25**(4), 397–414.

Sotskov, Y. N. (1991). Stability of an optimal schedule. *European Journal of Operational Research*, **55**(1), 91–102.

Sotskov, Y. N. and **A. Dolgui**, Stability radius of the optimal assembly line balance with fixed cycle time. *In Emerging Technologies and Factory Automation, 2001. Proceedings. 2001 8th IEEE International Conference on*. IEEE, 2001.

Sotskov, Y. N., **A. Dolgui**, and **M.-C. Portmann** (2006). Stability analysis of an optimal balance for an assembly line with fixed cycle time. *European Journal of Operational Research*, **168**(3), 783–797.

Sotskov, Y. N., **N. Egorova**, and **F. Werner** (2010). Minimizing total weighted completion time with uncertain data: A stability approach. *Automation and Remote Control*, **71**(10), 2038–2057.

Sotskov, Y. N., **N. G. Egorova**, **T.-C. Lai**, and **F. Werner**, The stability box in interval data for minimizing the sum of weighted completion times. *In SIMULTECH*. 2011.

Sotskov, Y. N. and **T.-C. Lai** (2012). Minimizing total weighted flow time under uncertainty using dominance and a stability box. *Computers & Operations Research*, **39**(6), 1271–1289.

Soyster, A. L. (1973). Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations research*, **21**(5), 1154–1157.

Van de Vonder, S., **E. Demeulemeester**, **W. Herroelen***, and **R. Leus** (2006). The trade-off between stability and makespan in resource-constrained project scheduling. *International Journal of Production Research*, **44**(2), 215–236.

Van Slyke, R. M. and **R. Wets** (1969). L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, **17**(4), 638–663.

Vieira, G. E., **J. W. Herrmann**, and **E. Lin** (2003). Rescheduling manufacturing systems: a framework of strategies, policies, and methods. *Journal of scheduling*, **6**(1), 39–62.

Vladimirou, H. and **S. A. Zenios** (1997). Stochastic linear programs with restricted recourse. *European Journal of Operational Research*, **101**(1), 177–192.

Von Neumann, J. and **O. Morgenstern** (1944). Theory of games and economic behavior.

Walker, W. E., **P. Harremoës**, **J. Rotmans**, **J. P. van der Sluijs**, **M. B. van Asselt**, **P. Janssen**, and **M. P. Krayer von Krauss** (2003). Defining uncertainty: a conceptual basis for uncertainty management in model-based decision support. *Integrated assessment*, **4**(1), 5–17.

Walker, W. E., **R. J. Lempert**, and **J. H. Kwakkel**, Deep uncertainty. *In Encyclopedia of operations research and management science*. Springer, 2013, 395–402.

Walker, W. E., **V. A. Marchau**, and **D. Swanson** (2010). Addressing deep uncertainty using adaptive policies: Introduction to section 2. *Technological Forecasting and Social Change*, **77**(6), 917–923.

Wang, D., **Z. Qin**, and **S. Kar** (2015). A novel single-period inventory problem with uncertain random demand and its application. *Applied Mathematics and Computation*, **269**, 133–145.

Wang, X. and **V.-A. Truong** (2015). Multi-priority online scheduling with cancellations. *Under review*.

Wierman, M. J. (2010). An introduction to the mathematics of uncertainty. *Creighton University*.

Williamson, O. E. (1993). Opportunism and its critics. *Managerial and decision economics*, **14**(2), 97–107.

Wojakowski, P. and **D. Warżołek** (2014). The classification of scheduling problems under production uncertainty. *Research in Logistics & Production*, **4**(3), 245–256.

Wu, S. D., **E.-S. Byeon**, and **R. H. Storer** (1999). A graph-theoretic decomposition of the job shop scheduling problem to achieve scheduling robustness. *Operations Research*, **47**(1), 113–124.

Wu, S. D., **R. H. Storer**, and **C. Pei-Chann** (1993). One-machine rescheduling heuristics with efficiency and stability as criteria. *Computers & Operations Research*, **20**(1), 1–14.

Xing, W. and **J. Zhang** (2000). Parallel machine scheduling with splitting jobs. *Discrete Applied Mathematics*, **103**(1), 259–269.

Xiong, J., **L.-n. Xing**, and **Y.-w. Chen** (2013). Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. *International Journal of Production Economics*, **141**(1), 112–126.

Xu, X., **W. Cui**, **J. Lin**, and **Y. Qian** (2013). Robust makespan minimisation in identical parallel machine scheduling problem with interval data. *International Journal of Production Research*, **51**(12), 3532–3548.

Xu, X., **J. Lin**, and **W. Cui** (2014). Hedge against total flow time uncertainty of the uniform parallel machine scheduling problem with interval data. *International Journal of Production Research*, **52**(19), 5611–5625.

Yang, B. and **J. Geunes** (2008). Predictive–reactive scheduling on a single resource with uncertain future jobs. *European Journal of Operational Research*, **189**(3), 1267–1283.

Yang, J. (2013). No tardiness rescheduling with order disruptions. *Industrial Engineering and Management Systems*, **12**(1), 51–62.

Yang, J. and **G. Yu** (2002). On the robust single machine scheduling problem. *Journal of Combinatorial Optimization*, **6**(1), 17–33.

Yin, Y., **T. Cheng**, and **D.-J. Wang** (2016). Rescheduling on identical parallel machines with machine disruptions to minimize total completion time. *European Journal of Operational Research*, **252**(3), 737–749.

You, F., **J. M. Pinto**, **I. E. Grossmann**, and **L. Megan** (2011). Optimal distribution-inventory planning of industrial gases. ii. minlp models and algorithms for stochastic cases. *Industrial & Engineering Chemistry Research*, **50**(5), 2928–2945.

Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, **8**(3), 338–353.

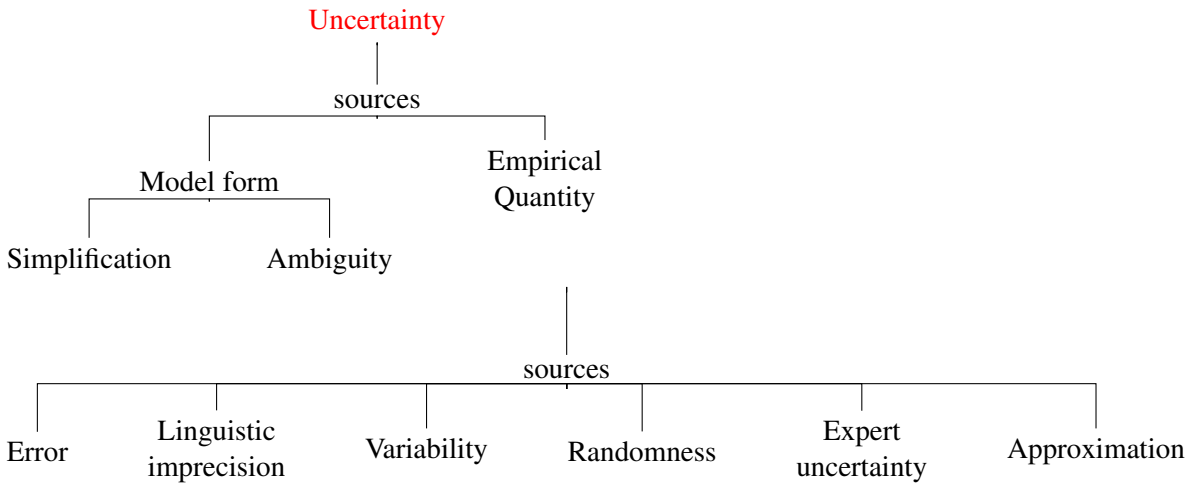
Zhou, S., **H. Chen**, **R. Xu**, and **X. Li** (2014). Minimising makespan on a single batch processing machine with dynamic job arrivals and non-identical job sizes. *International Journal of Production Research*, **52**(8), 2258–2274.

Zimmermann, H.-J. (1978). Fuzzy programming and linear programming with several objective functions. *Fuzzy sets and systems*, **1**(1), 45–55.

Zimmermann, H.-J. (2010). Fuzzy set theory. *Wiley Interdisciplinary Reviews: Computational Statistics*, **2**(3), 317–332.

Appendix A

Uncertainty sources

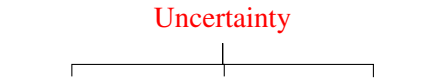


Uncertainty source	Meaning
Simplifications	Assumptions made to convert a complex model into a simpler model.
Ambiguity	Alternatives assumptions or techniques may be available for developing the model which may lead to alternative models.
Error	Error due to measurements or to subjective judgement
Linguistic imprecision	Quantities are not well specified.
Variability	Quantities that are variable over time.
Randomness	Probabilistic uncertainty.
Expert uncertainty	Divergent beliefs about the real value of the quantity.
Approximation	Deviation between the assumed value and the real value.

FIGURE A.1: Uncertainty classification by Morgan and Henrion (1990) according to the location of uncertainty

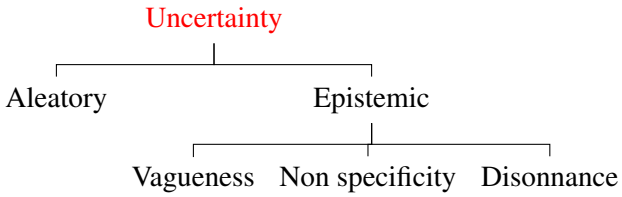
Appendix B

Uncertainty typologies



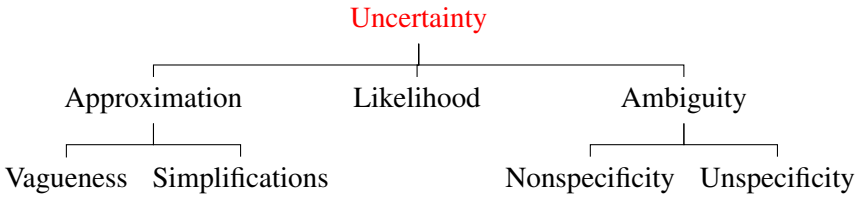
Uncertainty type	Meaning
Probability	Likelihood of future event
Ambiguity	Multiplicity of possible states for a single concept or event
Vagueness	Lack of fine-graded distinctions or boundaries, imprecision

FIGURE B.1: Uncertainty typology according to Smithson (1989, 1990, 2012)



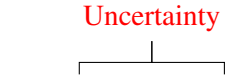
IUncertainty type	Meaning
Vagueness	Information are imprecisely defined, unclear, or indistinct
Non-specificity	Having multiple alternatives that are all possible but not specified
Dissonance	Existence of totally or partially conflicting evidence

FIGURE B.2: Uncertainty types by Oberkampf *et al.* (2004)



Uncertainty type	Meaning
Ambiguity	Having multioutcomes for processes or systems
Unspecificity	Outcomes are incompletely defined
Non-specificity	Outcomes are incorrectly defined
Approximations	Using vague semantics in language, approximate reasoning
Vagueness	Non crispness of belonging and non belonging of elements to a set
Simplifications	Assumptions needed to make problems tractable
Likelihood	Randomness and sampling

FIGURE B.3: Uncertainty types by Ayyub and Klir (2006)



uncertainty type	Meaning
Ambiguity	Multiplicity without preference to an alternative,
Fuzziness	lack of definite or sharp distinction

FIGURE B.4: Uncertainty types by Wierman (2010)

Appendix C

Scheduling example in the splitting case

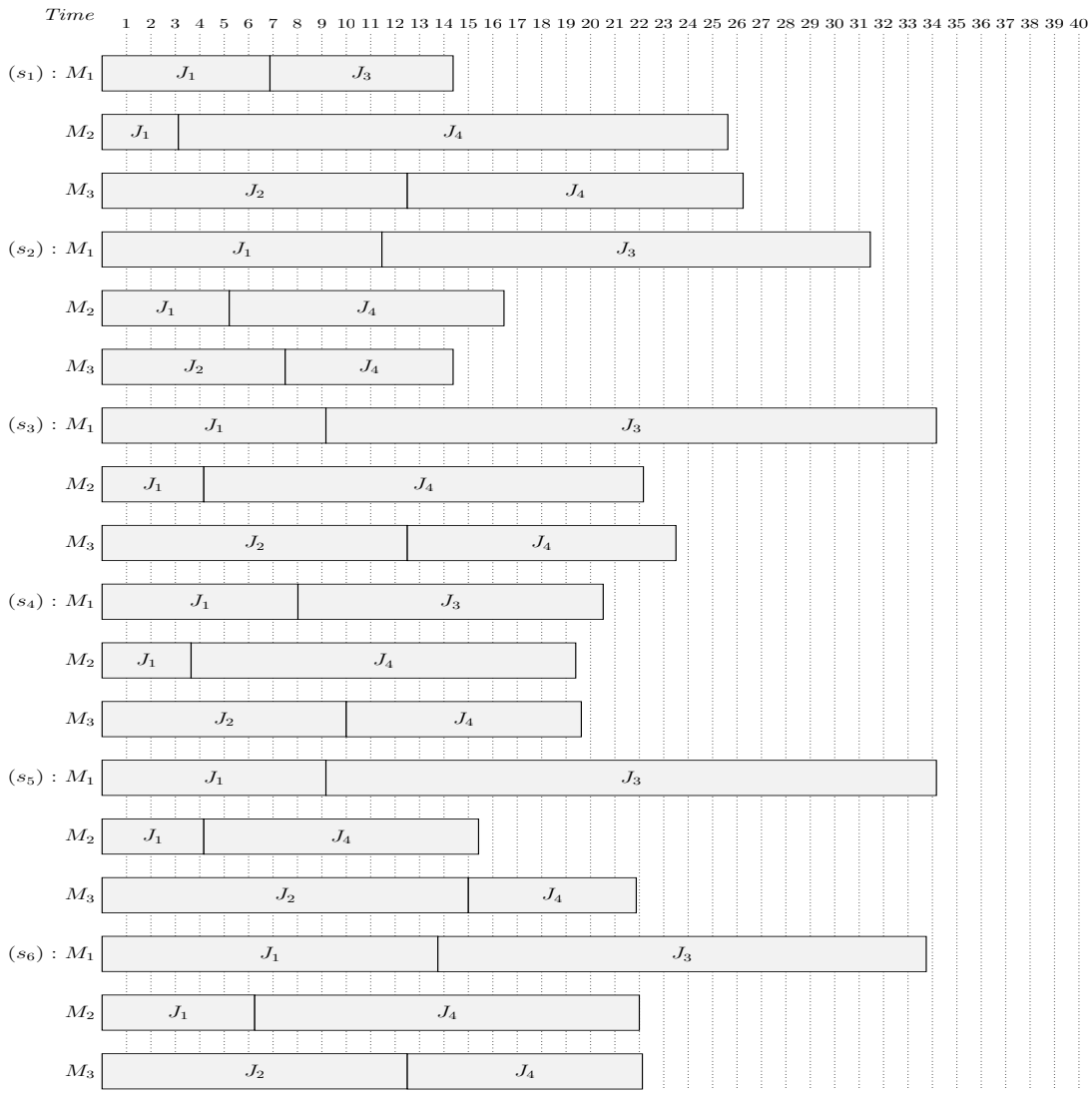


FIGURE C.1: The potential scenarios scheduled according to s_{max} in the splitting case

.

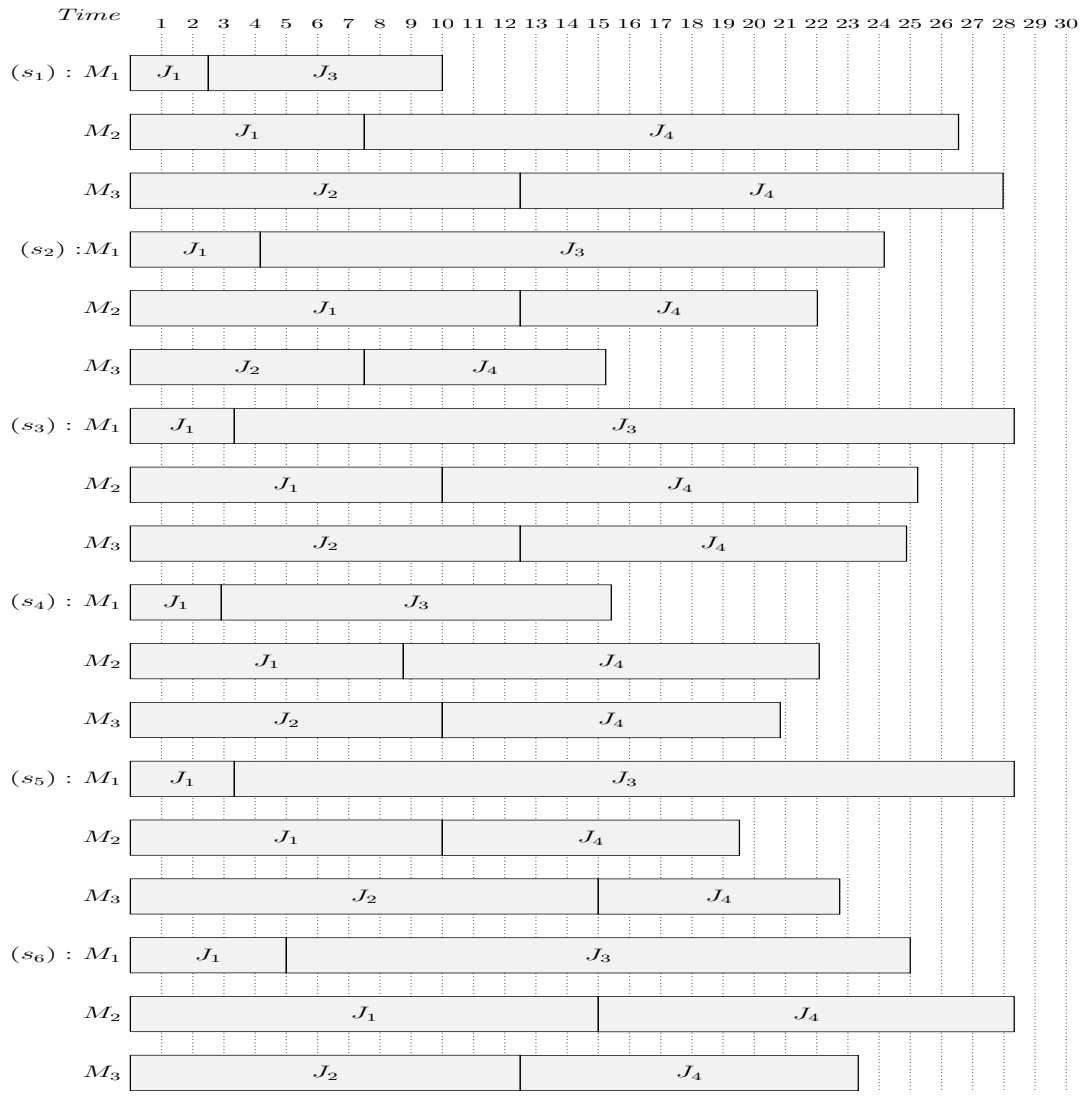


FIGURE C.2: The potential scenarios scheduled according to $s_{average}$ in the splitting case

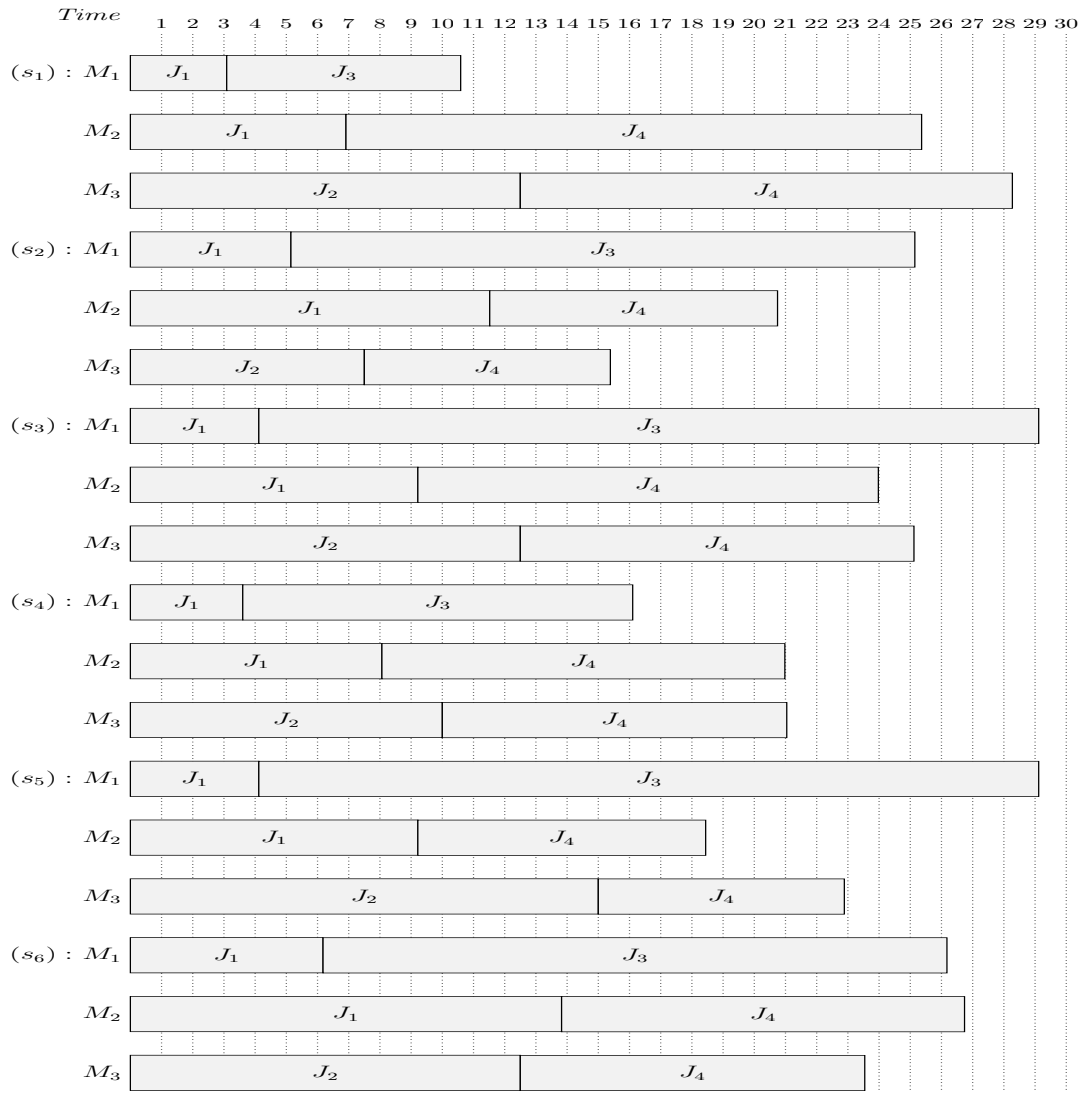


FIGURE C.3: The potential scenarios scheduled according to s_{median} in the splitting case

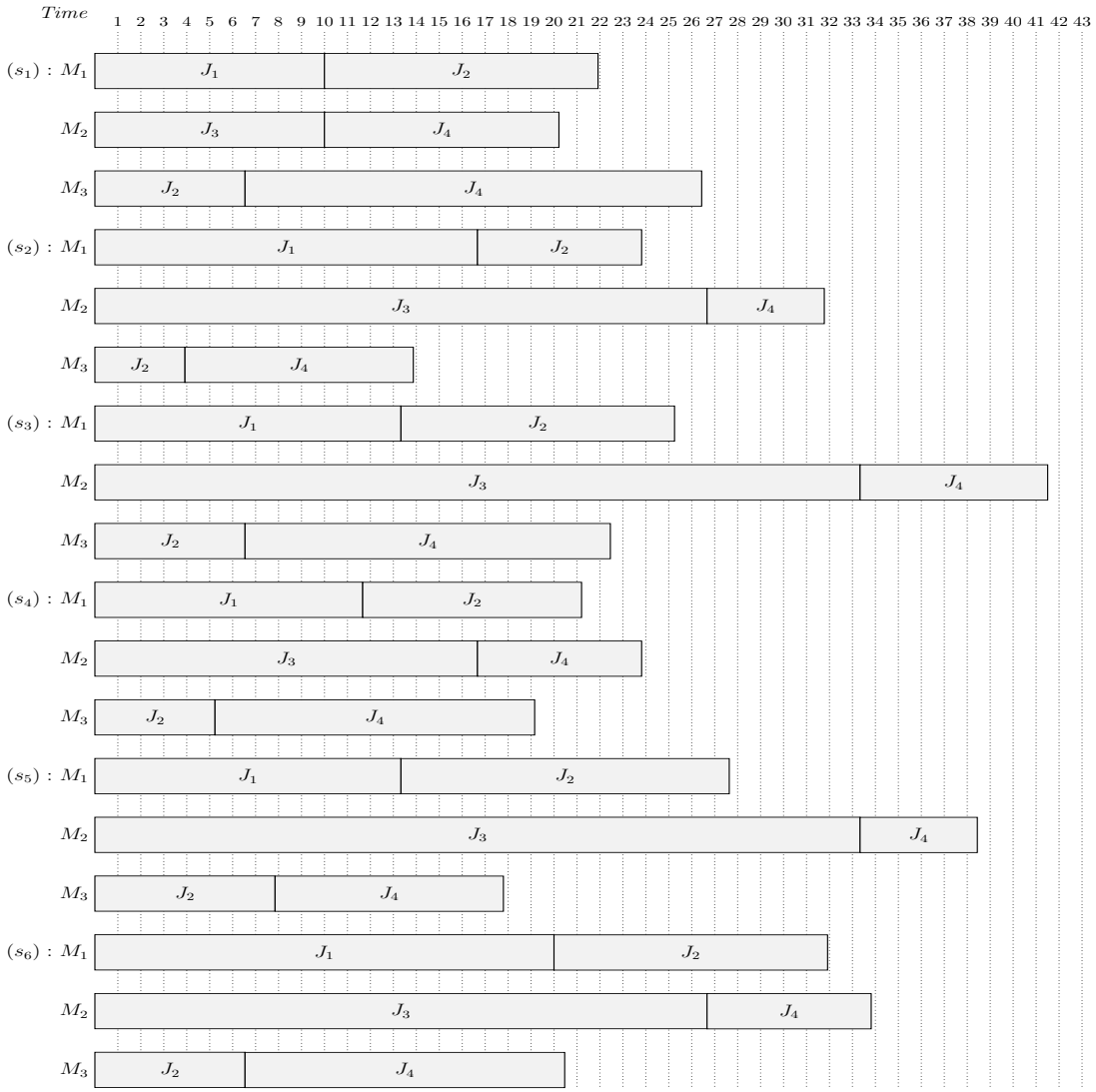


FIGURE C.4: The potential scenarios scheduled according to s_{min} in the splitting case

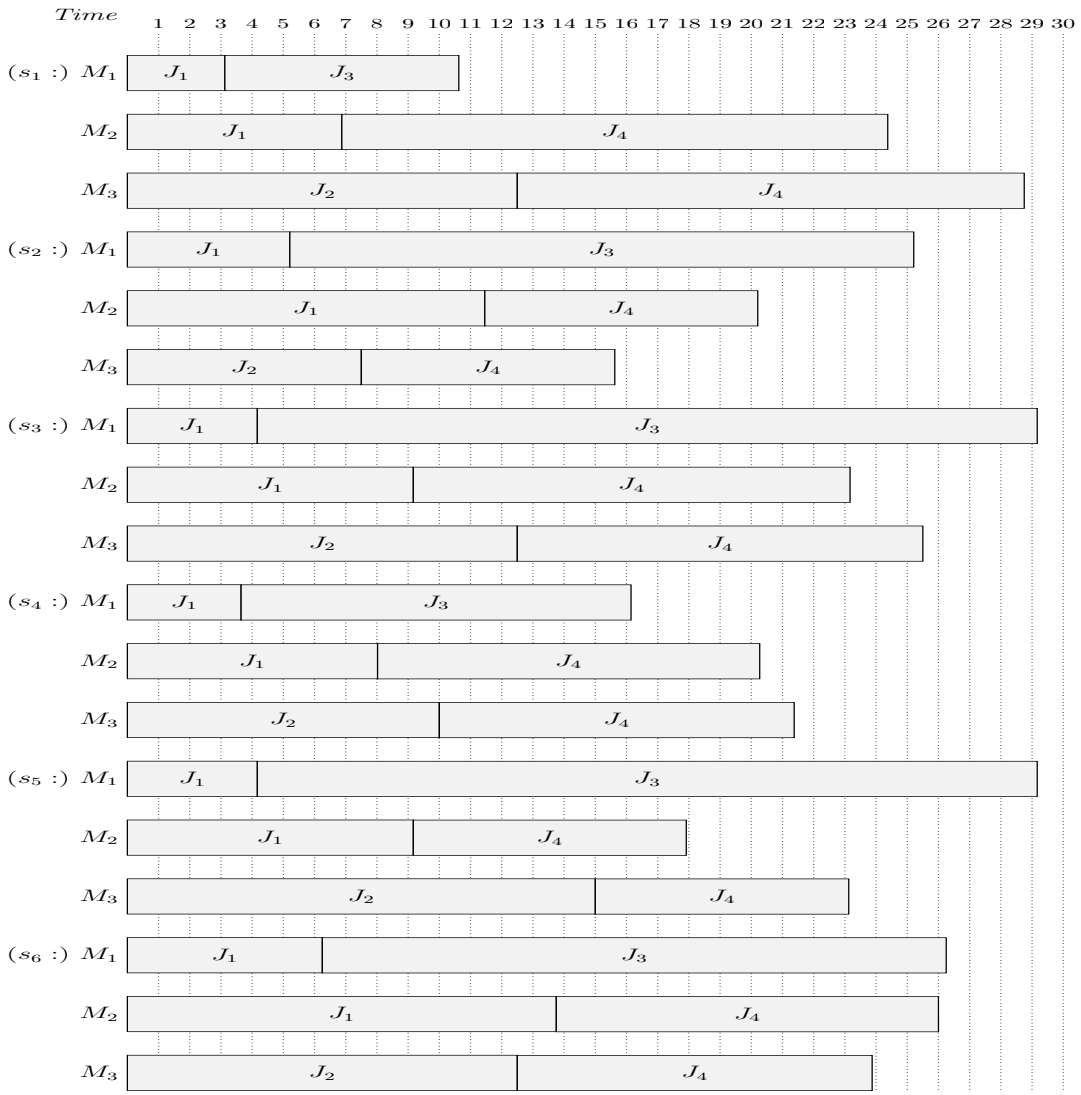
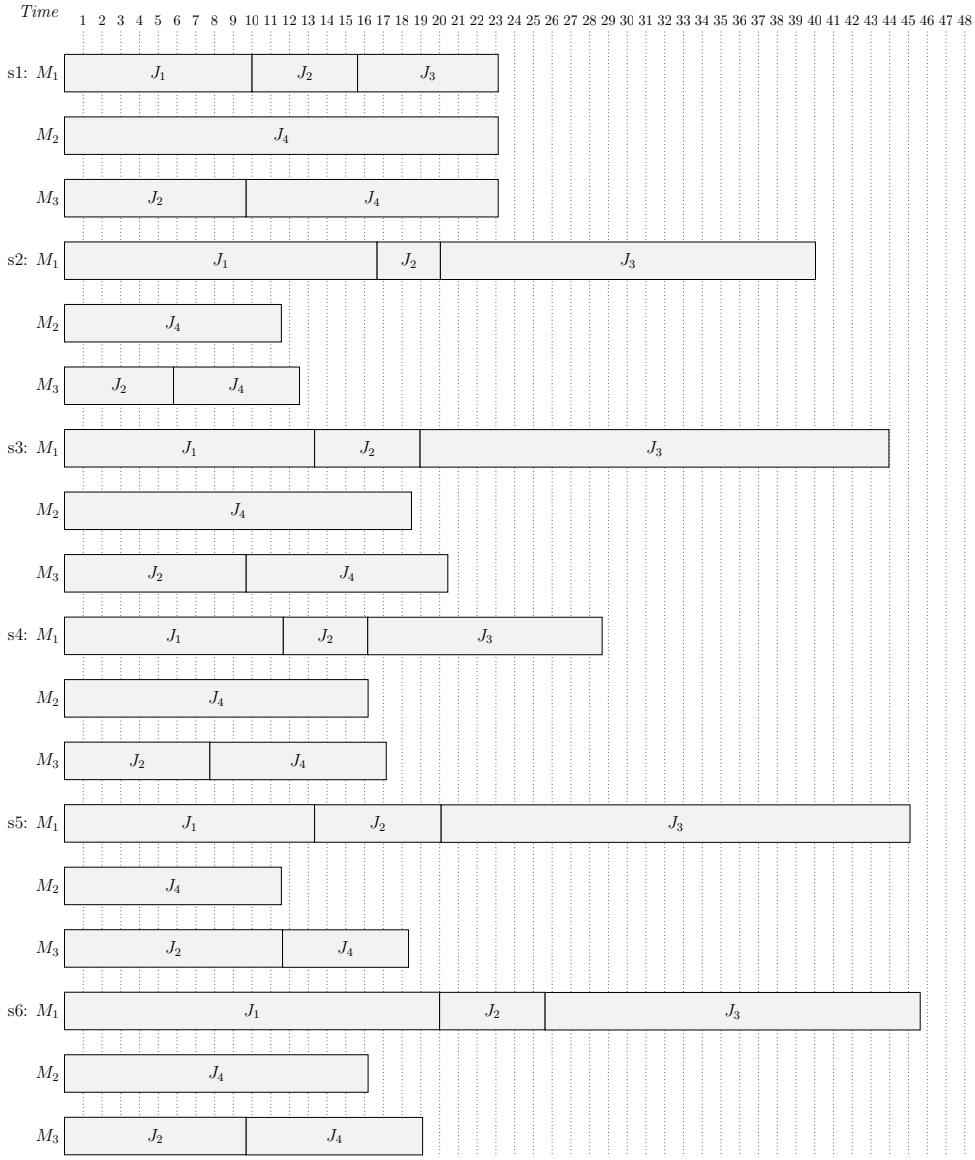


FIGURE C.5: The potential scenarios scheduled according to s_{random} in the splitting case

FIGURE C.6: Potential scenarios scheduled according to the nominal scenario solution in the preemptive case



Appendix D

Scheduling example in the preemptive case

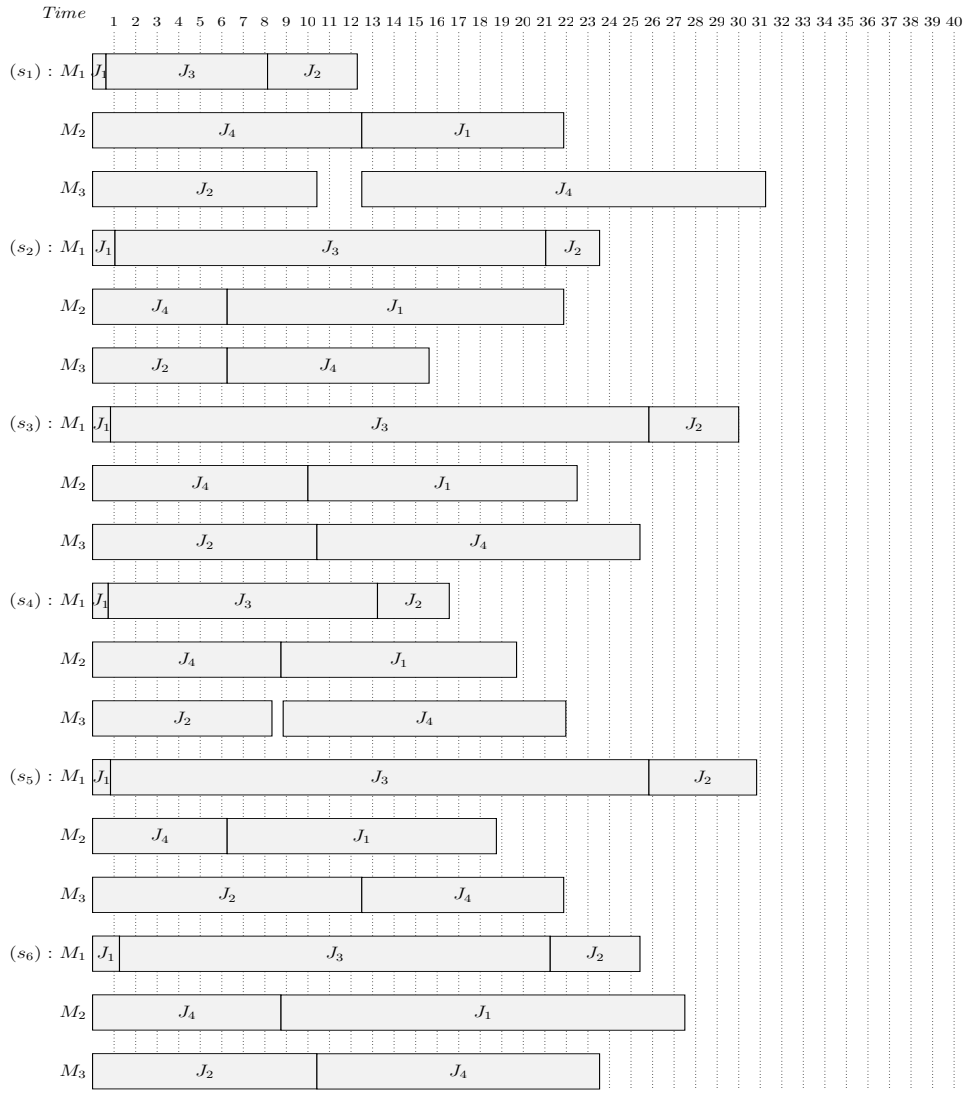


FIGURE D.1: The potential scenarios scheduled according to s_{max} in the preemptive case

.

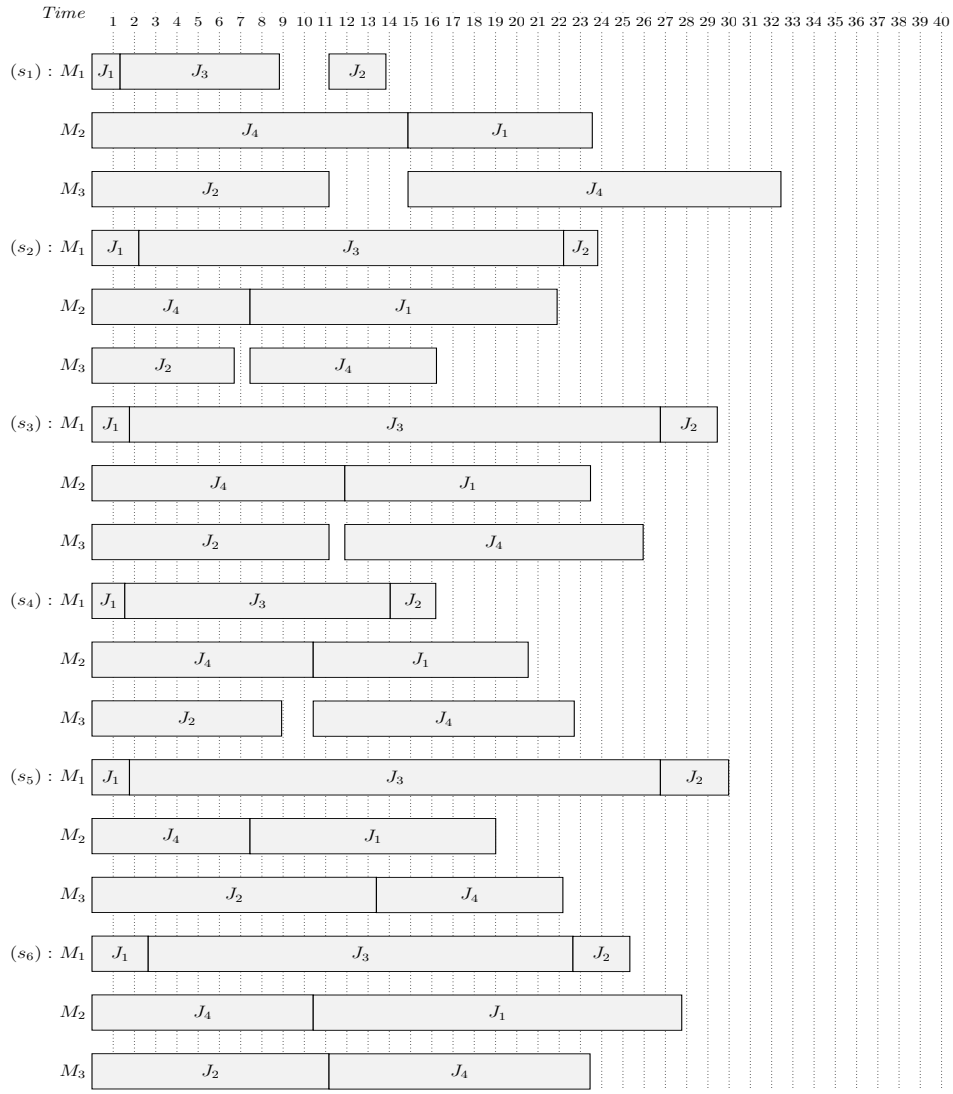


FIGURE D.2: The potential scenarios scheduled according to $s_{average}$ in the preemptive case

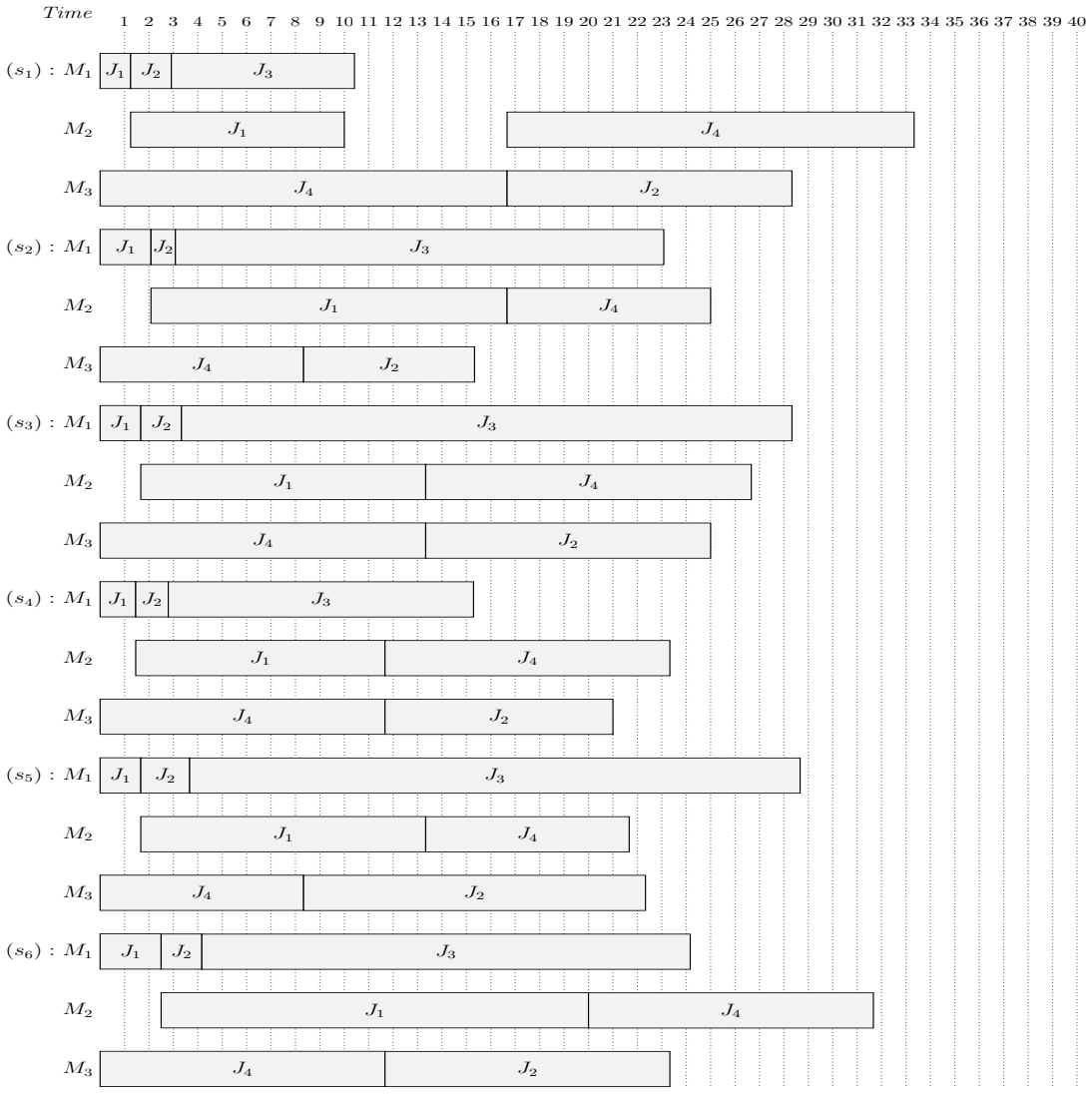


FIGURE D.3: The potential scenarios scheduled according to s_{median} in the preemptive case

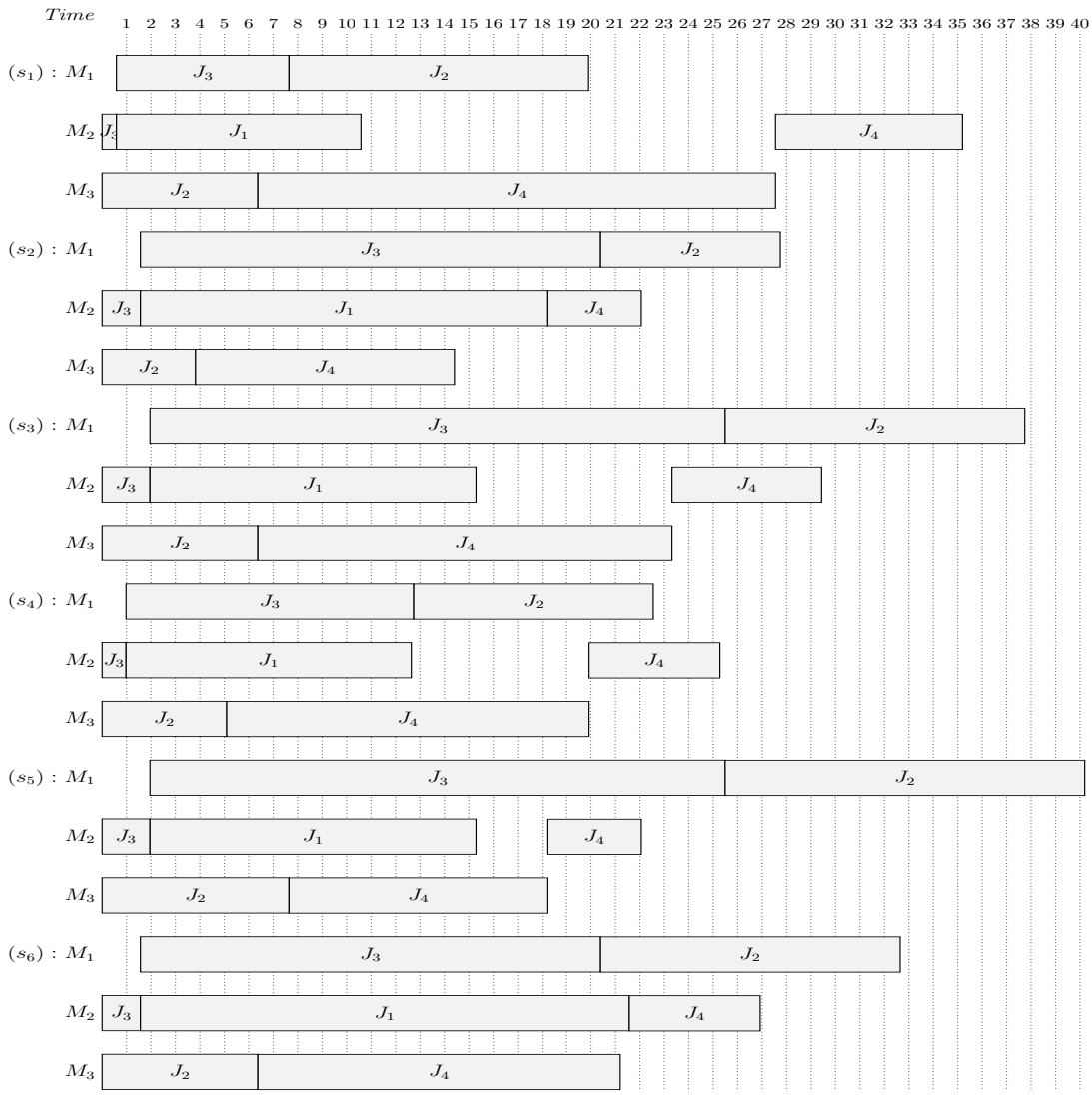


FIGURE D.4: The potential scenarios scheduled according to s_{min} in the preemptive case

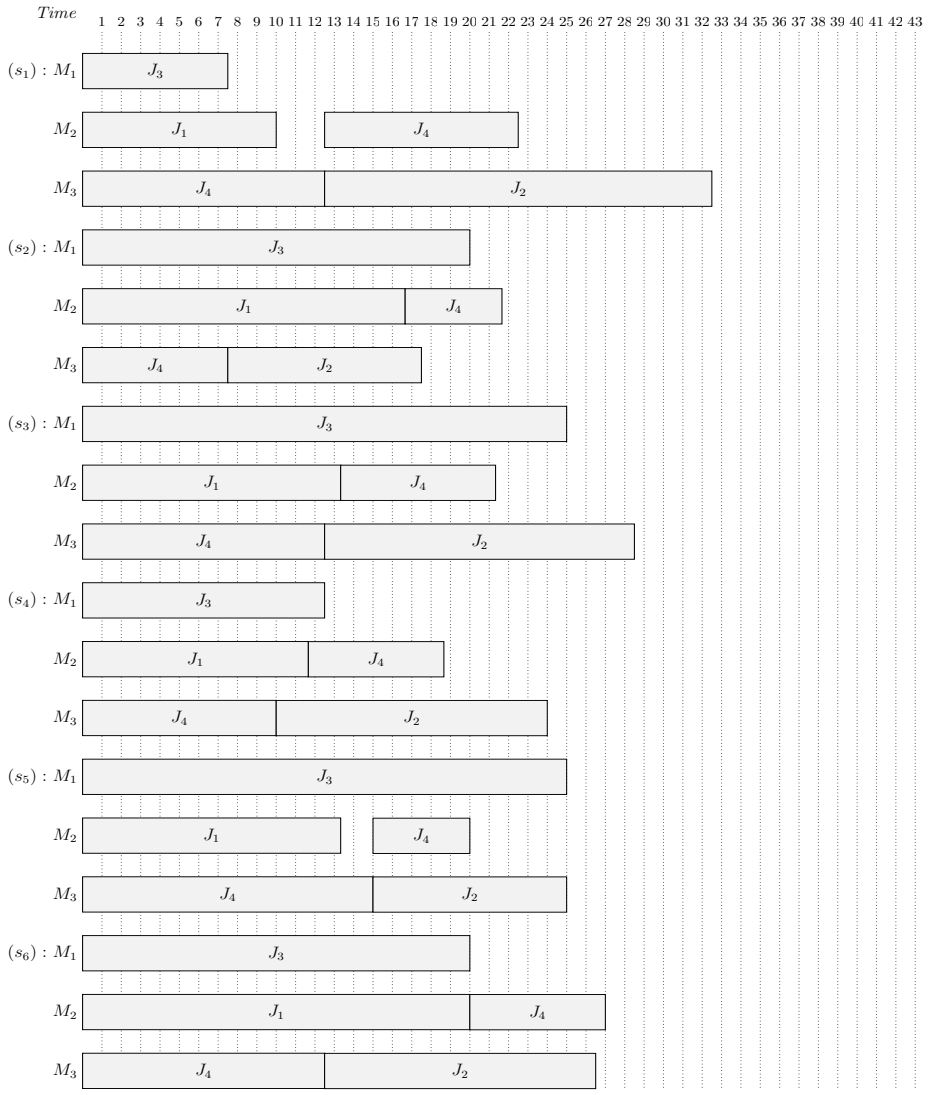
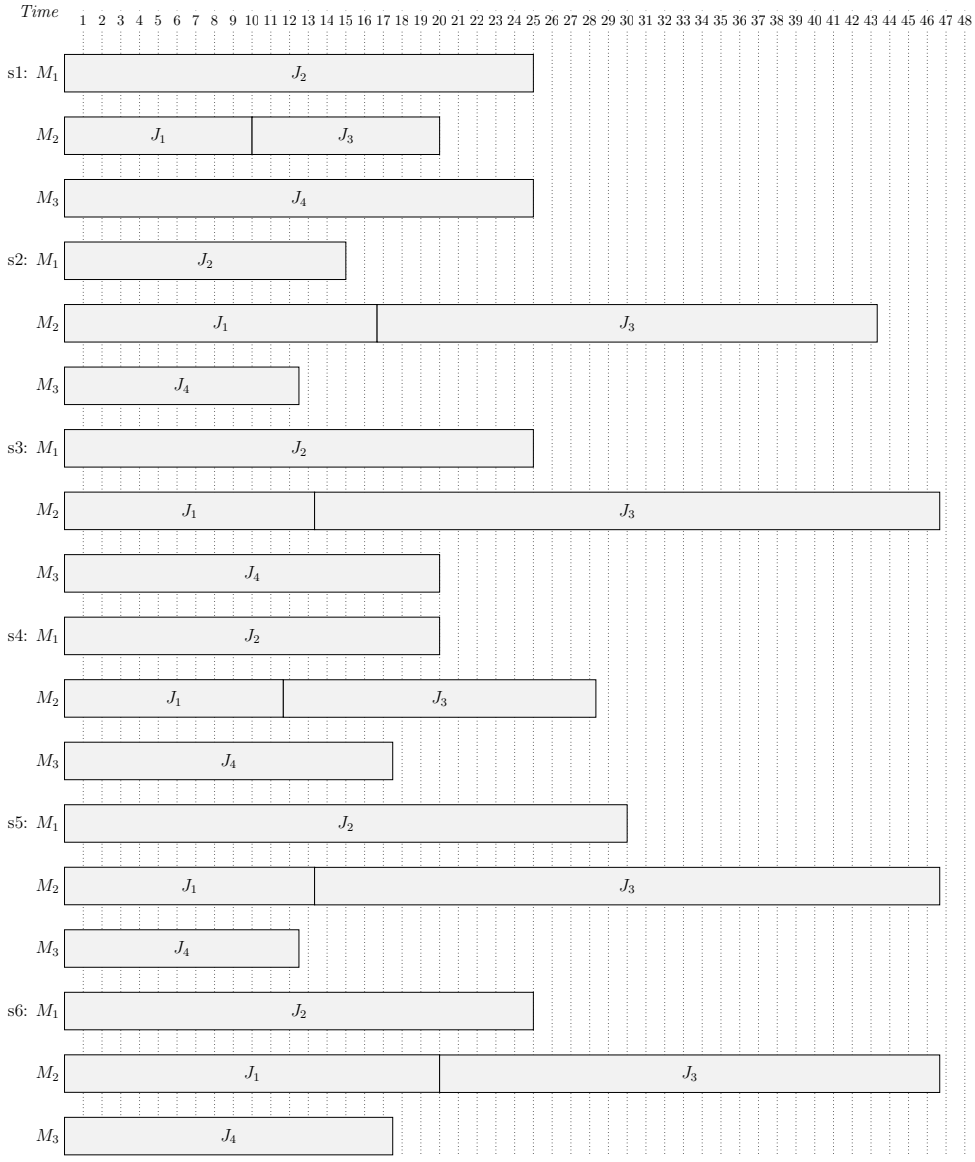


FIGURE D.5: The potential scenarios scheduled according to s_{random} in the preemptive case

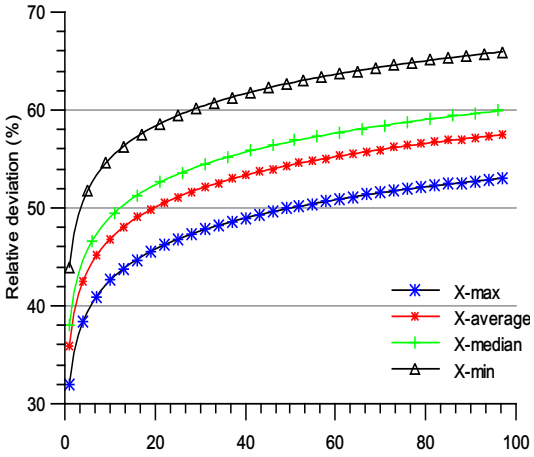
FIGURE D.6: Potential scenarios scheduled according to the nominal scenario solution in the preemptive case



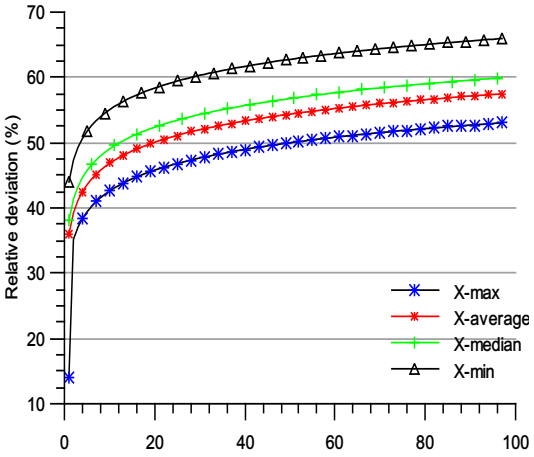
Appendix E

Logarithmic trend line functions

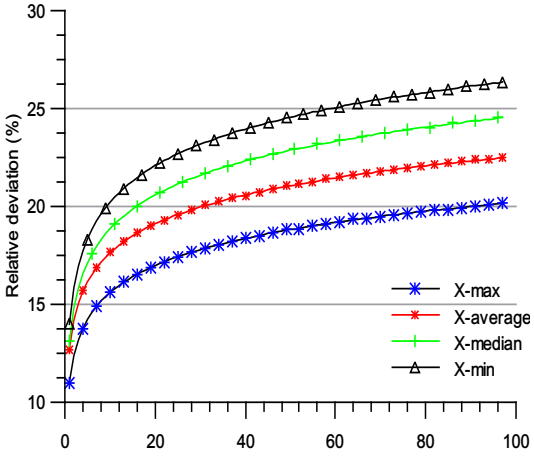
FIGURE E.1: Artificial scenario solution worst-case makespans in the preemptive case: medium size workshops under medium uncertainty



a. (m=7,n=10,d=2)



b. (m=7,n=50,d=2)

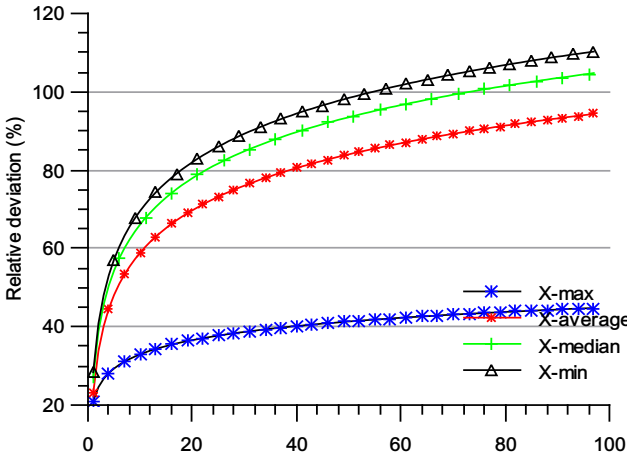


c. (m=7,n=200,d=2)

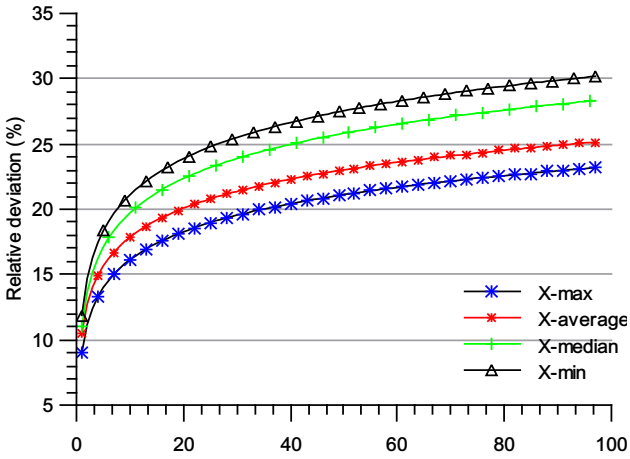
m=7	n	y	R^2
X_{max}	10	4.68ln(x) + 32.15	0.965
	50	4.36ln(x) + 17	0.979
	200	2.12ln(x) + 11.02	0.975
$X_{average}$	10	4.73ln(x) + 36.17	0.967
	50	4.38ln(x) + 19.2	0.963
	200	2.14ln(x) + 12.71	0.959
X_{median}	10	4.78n(x) + 48.12	0.967
	50	4.73ln(x) + 22.16	0.980
	200	2.51ln(x) + 13.18	0.976
X_{min}	10	4.96ln(x) + 45.05	0.959
	50	4.78ln(x) + 26.14	0.974
	200	2.74ln(x) + 14.12	0.954

d. logarithmic trends

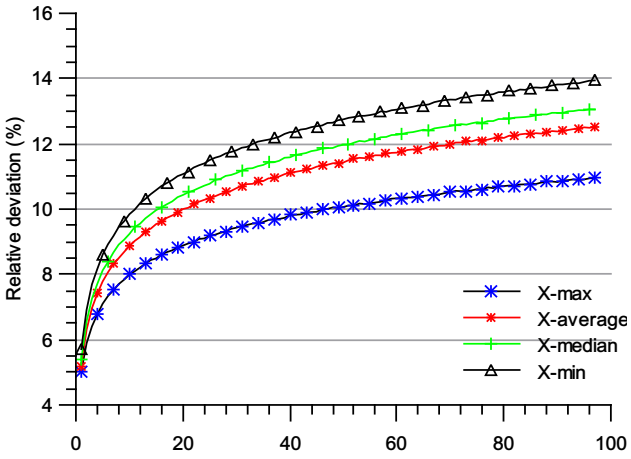
FIGURE E.2: Artificial scenario solution worst-case makespans in the preemptive case: small size workshops under medium uncertainty



a. (m=3,n=10,d=2)



b. (m=3,n=50,d=2)

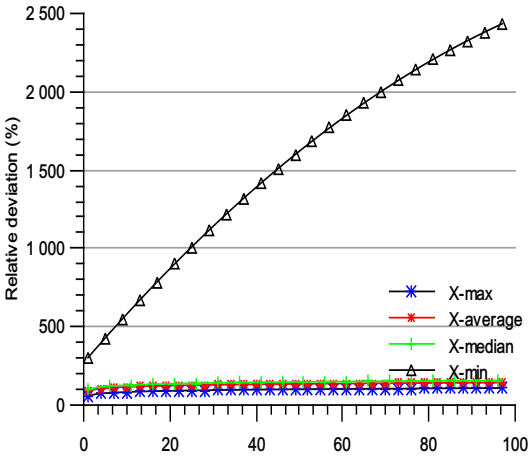


c. (m=3,n=200,d=2)

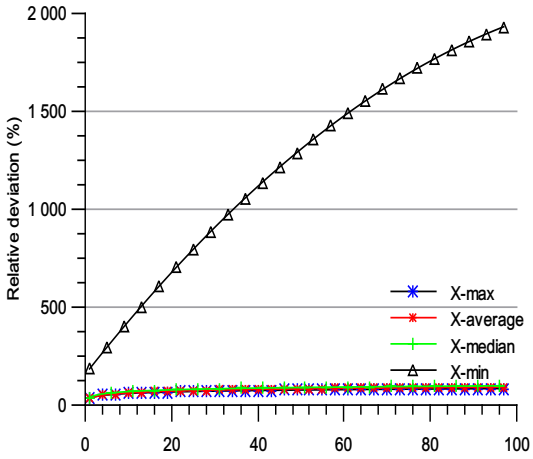
m=3	n	y	R^2
X_{max}	15	15.21ln(x) + 21.18	0.980
	50	3.26ln(x) + 9.12	0.953
	200	1.61ln(x) + 5.45	0.978
$X_{average}$	15	15.6ln(x) + 23	0.951
	50	3.24ln(x) + 10.53	0.962
	200	1.63ln(x) + 5.61	0.970
X_{median}	15	17.25ln(x) + 27.18	0.968
	50	3.79ln(x) + 11.30	0.957
	200	1.69ln(x) + 5.74	0.960
X_{min}	15	17.92ln(x) + 28.45	0.951
	50	4.75ln(x) + 11.96	0.976
	200	1.82ln(x) + 5.76	0.980

d. logarithmic trends

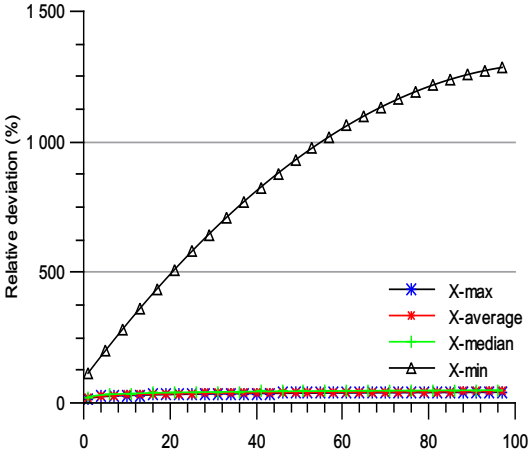
FIGURE E.3: Artificial scenario solution worst-case makespans in the preemptive case: medium size workshops under high uncertainty



a. (m=7,n=10,d=3)



a. (m=7,n=50,d=3)

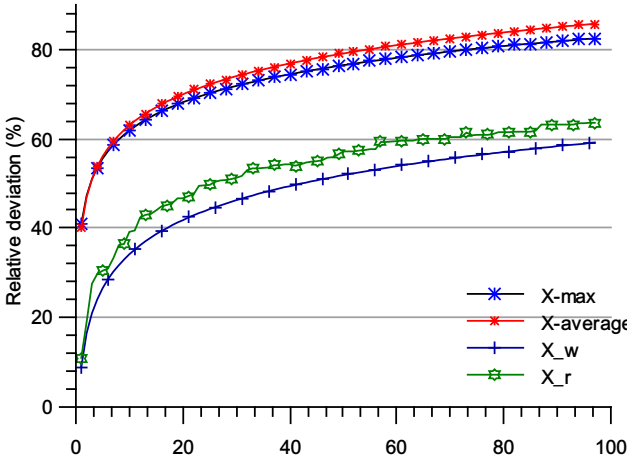


a. (m=7,n=200,d=3)

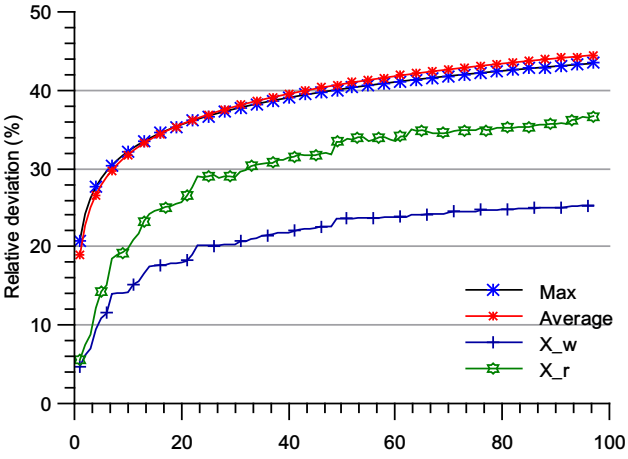
m=7	n	y	R^2
X_{max}	10	11.58ln(x) + 58.51	0.97
	50	10.43ln(x) + 34.16	0.98
	200	5.53ln(x) + 16.02	0.96
$X_{average}$	10	13.15ln(x) + 88.24	0.96
	50	12.1ln(x) + 32.04	0.95
	200	5.75ln(x) +18.34	0.95
X_{median}	10	13.46ln(x) + 97.17	0.96
	50	12.26ln(x) + 39.2	0.957
	200	5.92ln(x) + 22.15	0.96
X_{min}	10	-0.17x^2 +22x + 259.42	0.99
	50	-0.1272x^2 + 22x + 127.01	0.95
	200	-0.10x^2 + 22.6x + 92	0.95

d. line trend functions

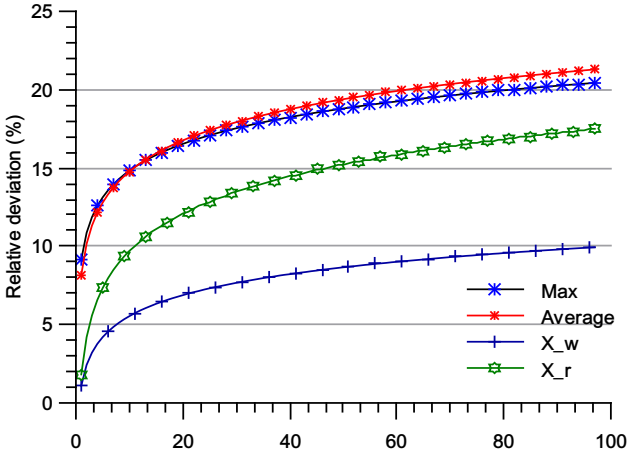
FIGURE E.4: Optimal robust solution worst-case makespans in the splitting case: small size workshops under high uncertainty



a. (m=3,n=10,d=3)



b. (m=3,n=50,d=3)

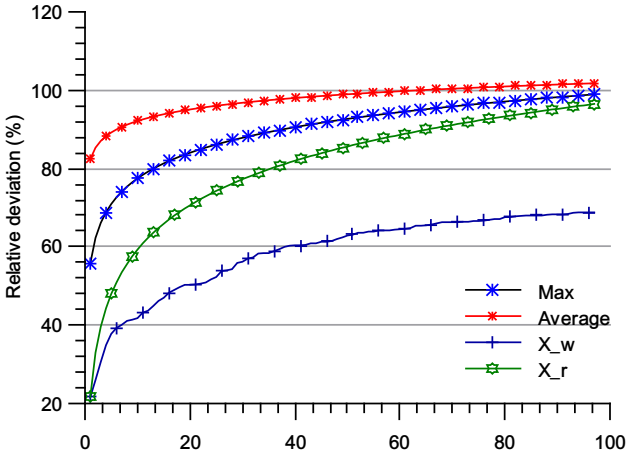


c. (m=3,n=200,d=3)

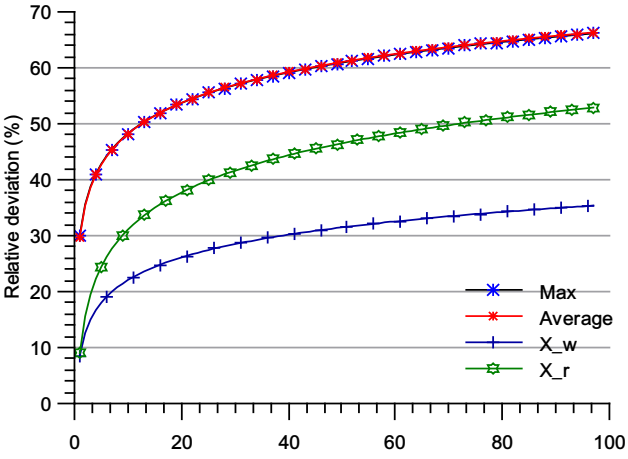
m=3	*n*	*y*	R^2
X_{max}	10	9.86ln(x) + 40.27	0.97
	50	5.59ln(x) + 18.89	0.97
	200	2.46ln(x) + 9.15	0.97
$X_{average}$	10	9.27ln(x) + 41.60	0.950
	50	4.94ln(x) + 20.78	0.97
	200	2.86ln(x) + 8.16	0.96
X_w	10	11ln(x) + 8.86	0.95
	50	4.85ln(x) + 3.76	0.95
	200	1.93ln(x) + 1.08	0.95
X_r	10	11.12ln(x) + 13.28	0.974
	50	7.26ln(x) + 4.03	0.96
	200	3.42ln(x) + 1.82	0.95

d. logarithmic line trend functions

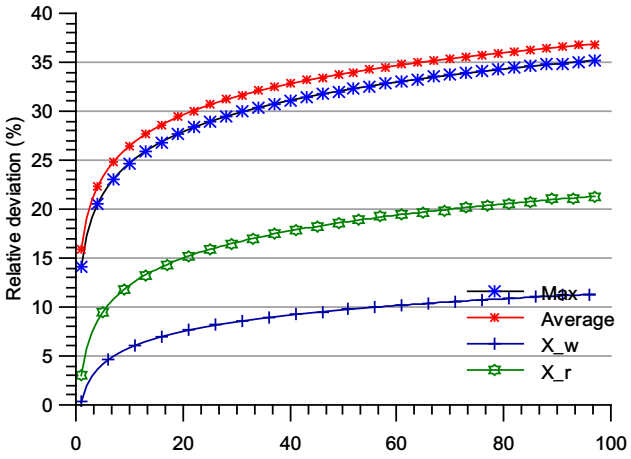
FIGURE E.5: Optimal robust solution worst-case makespans in the splitting case: medium size workshops under high uncertainty



a. (m=7,n=10,d=3)



b. (m=7,n=50,d=3)

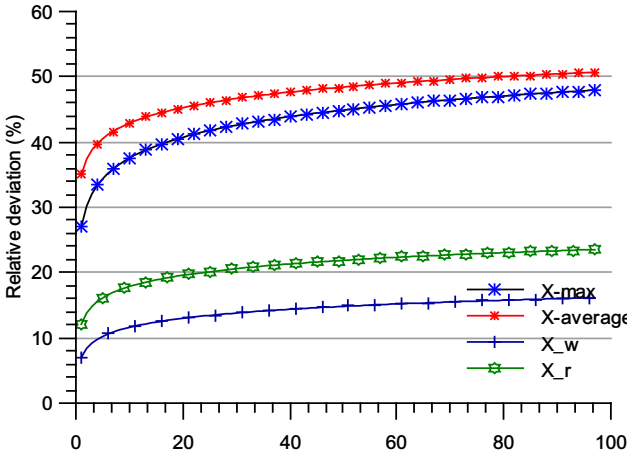


c. (m=7,n=200,d=3)

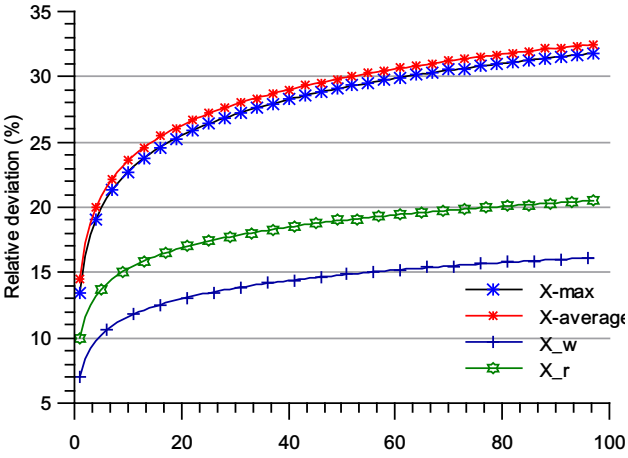
m=7	*n*	*y*	R^2
X_{max}	10	9.43ln(x) + 55.75	0.97
	50	7.87ln(x) + 30.14	0.98
	200	4.62ln(x) + 14.05	0.96
$X_{average}$	10	4.23ln(x) + 82.39	0.96
	50	7.97ln(x) + 29.83	0.95
	200	4.58ln(x) + 15.89	0.95
X_w	10	11.32ln(x) + 17.73	0.96
	50	5.85ln(x) + 8.57	0.96
	200	2.41ln(x) + 1.27	0.96
X_r	10	16.36ln(x) + 21.63	0.96
	50	9.59ln(x) + 9	0.96
	200	4.1ln(x) + 3.01	0.96

d. logarithmic line trend functions

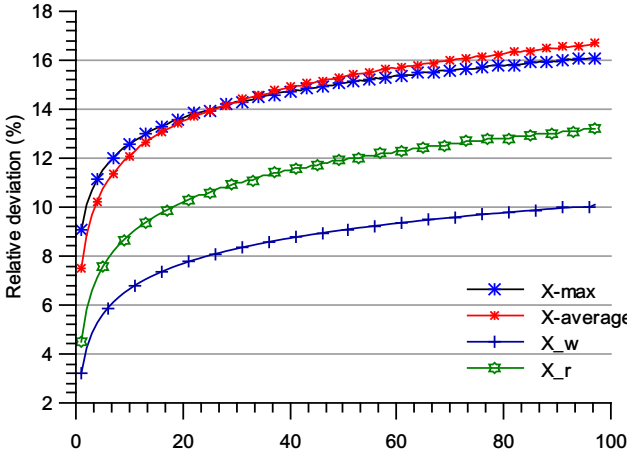
FIGURE E.6: Optimal robust solution worst-case makespans in the splitting case: medium size workshops under medium uncertainty



a. (m=7,n=10,d=2)



b. (m=7,n=50,d=2)

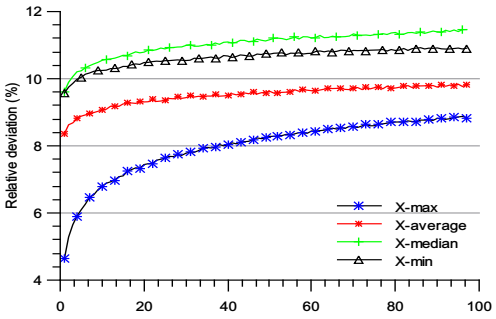


c. (m=7,n=200,d=2)

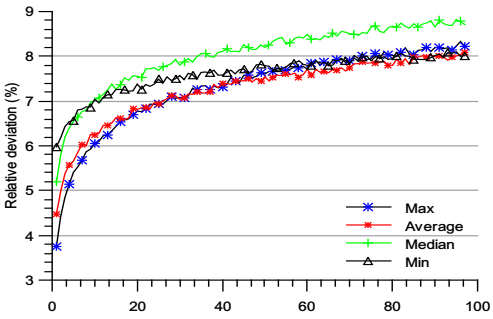
m=7	n	y	R^2
X_{max}	10	4.58ln(x) + 27.00	0.96
	50	3.99ln(x) + 13.51	0.97
	200	1.54ln(x) + 9.01	0.97
$X_{average}$	10	3.43ln(x) + 34.98	0.96
	50	3.93ln(x) + 14.51	0.96
	200	2.00ln(x) + 7.49	0.95
X_w	10	2ln(x)+7	0.954
	50	1.8ln(x)+3.7	0.97
	200	1.5ln(x)+3.2	0.96
X_r	10	2.5ln(x)+12.1	0.96
	50	2.3ln(x)+10.1	0.97
	200	1.9ln(x)+4.5	0.96

d. logarithmic line trend functions

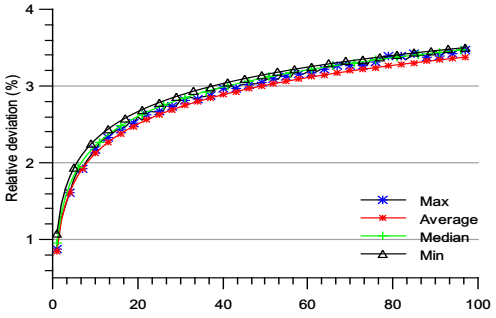
FIGURE E.7: Artificial scenario solution worst-case makespans in the splitting case: high size workshops under low uncertainty



a. (m=15,n=15,d=1)



b. (m=15,n=50,d=1)

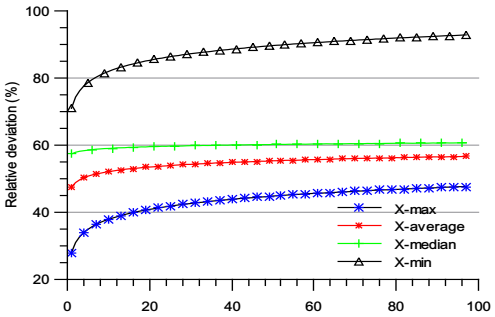


c. (m=15,n=200,d=1)

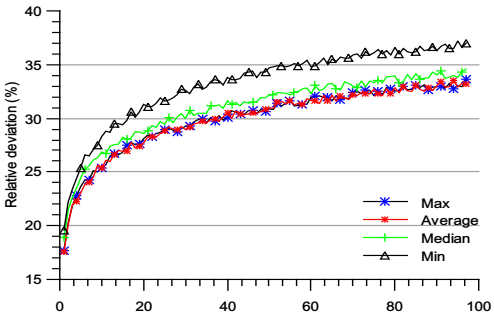
m=15	n	y	R^2
X_{max}	15	0.31ln(x) + 8.35	0.96
	50	0.92ln(x) + 4.64	0.96
	200	0.57ln(x) + 0.84	0.97
$X_{average}$	15	0.31ln(x) + 8.35	0.97
	50	0.77ln(x) + 4.46	0.96
	200	0.55ln(x) + 0.84	0.96
X_{median}	15	0.39ln(x) + 9.62	0.97
	50	0.77ln(x) + 5.25	0.97
	200	0.52ln(x) + 1.08	0.96
X_{min}	10	0.29ln(x) + 9.56	0.98
	50	0.77ln(x) + 5.89	0.98
	200	0.52ln(x) + 1.08	0.97

d. logarithmic line trend functions

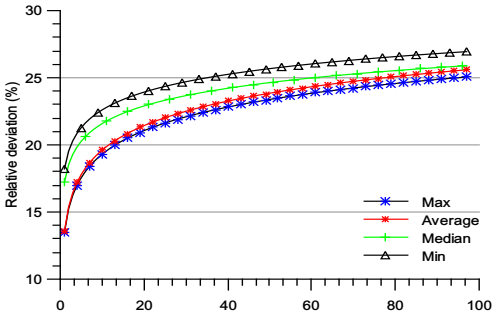
FIGURE E.8: Artificial scenario solution worst-case makespans under splitting: high size workshops under medium uncertainty



a. (m=15,n=15,d=2)



b. (m=15,n=50,d=2)

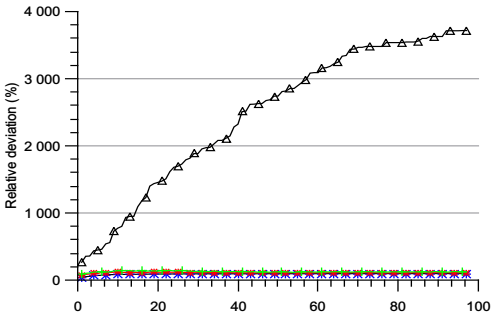


c. (m=15,n=200,d=2)

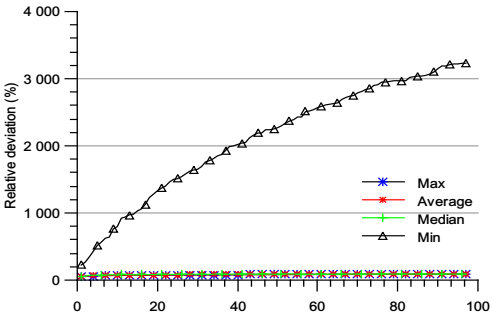
m=15	n	y	R^2
X_{max}	15	4.29ln(x) + 28.01	0.97
	50	3.35ln(x) + 17.85	0.97
	200	2.39ln(x) + 13.72	0.96
$X_{average}$	15	1.99ln(x) + 47.51	0.95
	50	3.35ln(x) + 17.85	0.96
	200	2.37ln(x) + 14.07	0.97
X_{median}	15	0.57ln(x) + 57.78	0.98
	50	3.35ln(x) + 17.8	0.96
	200	2.36ln(x) + 13.53	0.96
X_{min}	15	4.77ln(x) + 70.98	0.96
	50	3.35ln(x) + 18.85	0.97
	200	2.97ln(x) + 11.63	0.97

d. logarithmic line trend functions

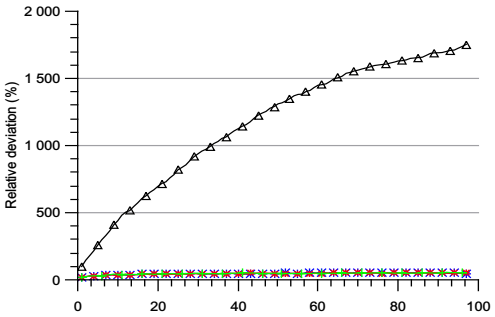
FIGURE E.9: Artificial scenario solution worst-case makespans under splitting: high size workshops under high uncertainty



a. (m=15,n=15,d=3)



b. (m=15,n=50,d=3)

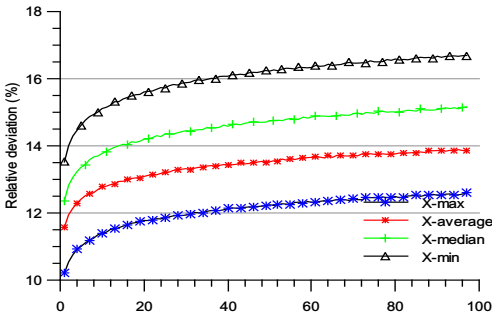


c. (m=15,n=200,d=3)

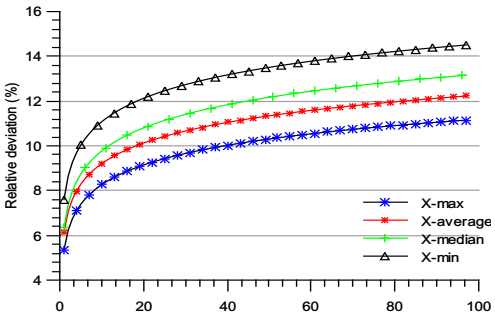
m=15	*n*	*y*	R^2
X_{max}	15	9.71ln(x) + 56.73	0.97
	50	7.91ln(x) + 50.99	0.97
	200	6.97ln(x) + 20.95	0.96
$X_{average}$	15	2.62ln(x) + 101.01	0.97
	50	7.39ln(x) + 55.84	0.97
	200	6.87ln(x) + 20.84	0.96
X_{median}	15	3.69ln(x) + 124.71	0.97
	50	6.69ln(x) + 62.61	0.97
	200	6.52ln(x) + 22.85	0.960
X_{min}	15	-0.34x^2 + 70.201x + 122.19	0.97
	50	-0.22x^2 + 51.682x + 302.32	0.97
	200	-0.15x^2 + 31.439x + 120.98	0.96

d. line trend functions

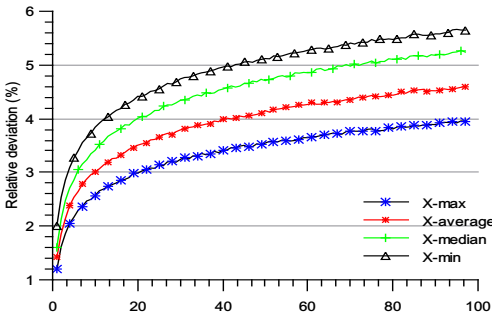
FIGURE E.10: Artificial scenario solution worst-case makespans in the preemptive case: high size workshops under low uncertainty



a. (m=15,n=15,d=1)



b. (m=15,n=50,d=1)

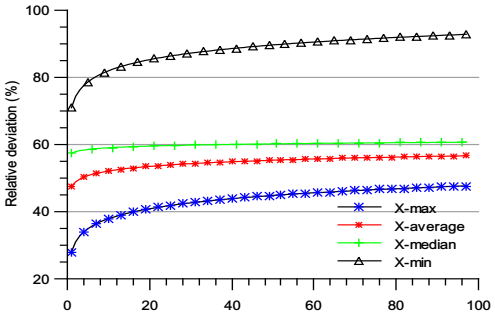


c. (m=15,n=200,d=1)

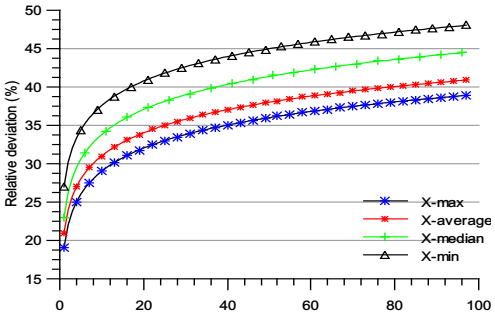
m=15	n	y	R^2
X_{max}	15	0.52ln(x) + 10.2	0.965
	50	1.27ln(x) + 5.34	0.954
	200	0.61ln(x) + 1.2	0.957
$X_{average}$	15	0.54ln(x) +11.6	0.979
	50	1.33ln(x) + 6.14	0.969
	200	0.7ln(x) + 1.42	0.965
X_{median}	15	0.61ln(x) + 12.46	0.974
	50	1.49ln(x) + 6.34	0.975
	200	0.89ln(x) + 1.64	0.966
X_{min}	15	0.77ln(x) + 13.56	0.980
	50	1.51ln(x) + 7.6	0.958
	200	0.87ln(x) + 2.06	0.953

d. logarithmic line trend functions

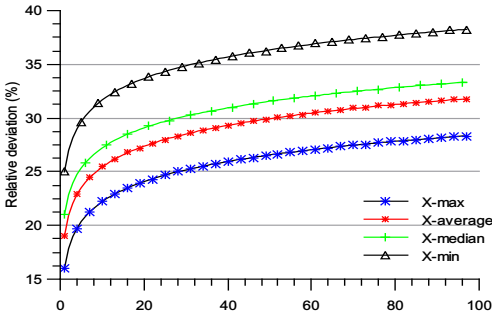
FIGURE E.11: Artificial scenario solution worst-case makespans in the preemptive case: high size workshops under medium uncertainty



a. (m=15,n=15,d=2)



b. (m=15,n=50,d=2)

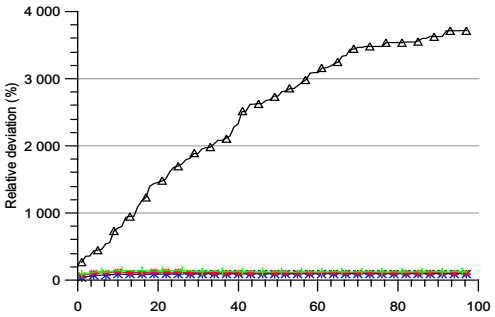


c. (m=15,n=200,d=2)

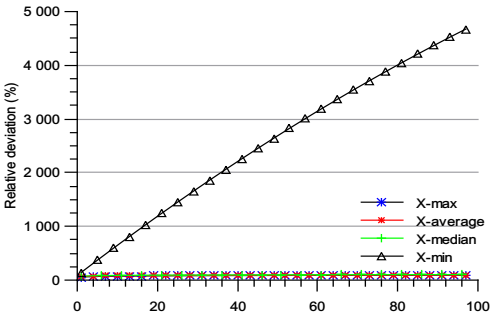
m=15	*n*	*y*	R^2
X_{max}	15	4.7ln(x)+29.06	0.977
	50	4.35ln(x)+19.34	0.973
	200	2.73ln(x)+16.43	0.961
$X_{average}$	15	4.9ln(x)+32.14	0.952
	50	4.58ln(x)+21.34	0.968
	200	2.8ln(x)+21.04	0.953
X_{median}	15	4.7ln(x)+34.21	0.953
	50	4.92ln(x)+23.15	0.951
	200	2.78 ln(x)+23.5	0.961
X_{min}	15	4.61ln(x) + 38.27	0.952
	50	4.68ln(x)+27.48	0.979
	200	2.91ln(x) + 25.37	0.979

d. logarithmic line trend functions

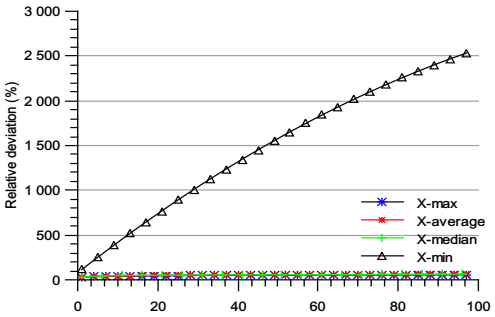
FIGURE E.12: Artificial scenario solution worst-case makespans in the preemptive case: high size workshops under high uncertainty



a. (m=15,n=15,d=3)



b. (m=15,n=50,d=3)

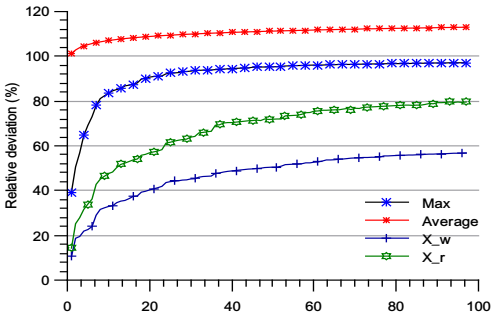


c. (m=15,n=200,d=3)

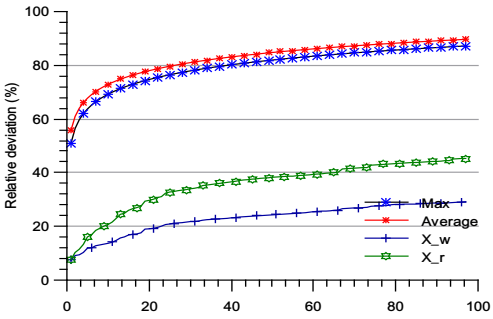
m=15	*n*	*y*	R^2
X_{max}	15	10.16ln(x) + 60.14	0.977
	50	8.10ln(x) + 55.04	0.973
	200	7.12ln(x) + 24.20	0.966
$X_{average}$	15	7.30ln(x) + 105.10	0.973
	50	6.82ln(x) + 65.10	0.978
	200	6.44ln(x) + 28.4	0.964
X_{median}	15	7.97ln(x) + 130.21	0.979
	50	7.12ln(x) + 73.5	0.970
	200	6.9ln(x) + 30.15	0.960
X_{min}	15	-0.3x^2 + 77x + 117	0.979
	50	-0.1x^2 + 57x + 87	0.970
	200	-0.1x^2 + 35x + 79	0.960

d. line trend functions

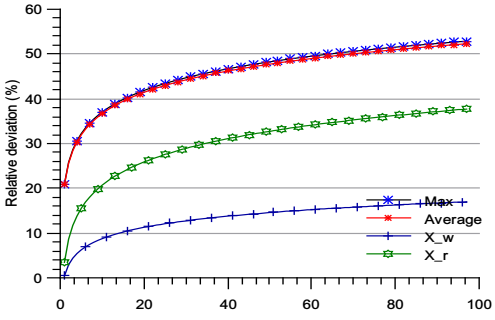
FIGURE E.13: Optimal robust solution worst-case makespans in the splitting case: high size workshops under high uncertainty



a. (m=15,n=15,d=3)



b. (m=15,n=50,d=3)

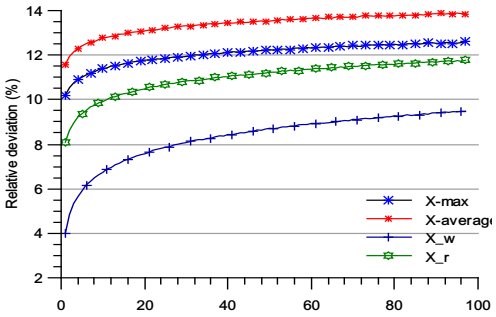


c. (m=15,n=200,d=3)

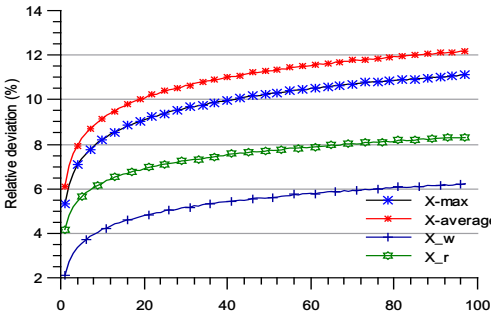
m=15	*n*	*y*	R^2
X_{max}	15	9.717ln(x) + 56.735	0.977
	50	7.9134ln(x) + 50.998	0.973
	200	6.9797ln(x) + 20.954	0.966
$X_{average}$	15	2.6236ln(x) + 101.01	0.973
	50	7.3924ln(x) + 55.847	0.978
	200	6.8744ln(x) + 20.849	0.964
X_w	15	10.847ln(x) + 8.1027	0.979
	50	5.7306ln(x) + 8.2735	0.970
	200	3.619ln(x) + 0.4162	0.960
X_r	15	14.983ln(x) + 12.956	0.979
	50	9.5062ln(x) + 9.217	0.970
	200	7.488ln(x) + 3.4519	0.960

d. logarithmic line trend functions

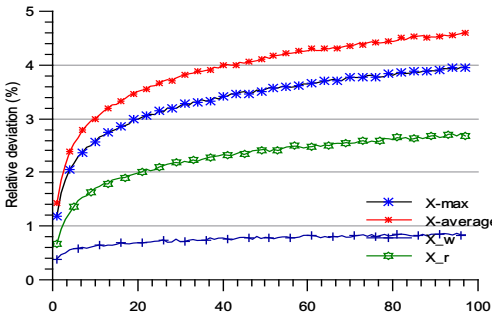
FIGURE E.14: Optimal robust solution worst-case makespans in the splitting case: high size workshops under low uncertainty



a. (m=15,n=15,d=1)



b. (m=15,n=50,d=1)

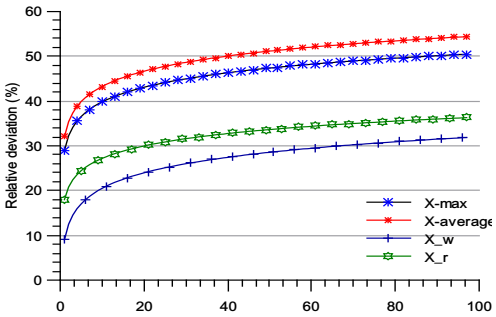


c. (m=15,n=200,d=1)

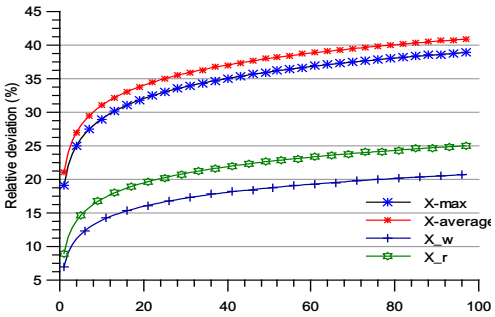
m=15	*n*	*y*	R^2
X_{max}	10	0.52ln(x) + 10.2	0.965
	50	1.27ln(x) + 5.34	0.954
	200	0.61ln(x) + 1.2	0.957
$X_{average}$	10	0.54ln(x) +11.6	0.979
	50	1.33ln(x) + 6.14	0.969
	200	0.7ln(x) + 1.42	0.965
X_w	15	1.58ln(x) +4.10	0.974
	50	0.91ln(x) + 2.1	0.975
	200	0.1ln(x)+0.4	0.966
X_r	15	0.8ln(x) +7.2	0.980
	50	0.98ln(x) + 4.2	0.958
	200	0.45ln(x) + 0.65	0.953

d. logarithmic line trend functions

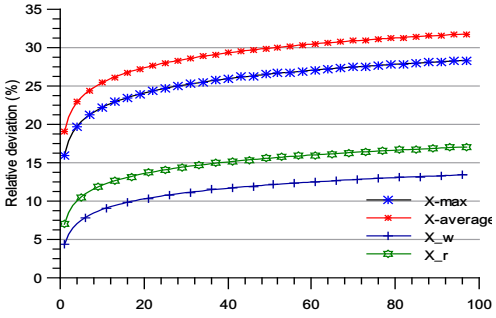
FIGURE E.15: Optimal robust solution worst-case makespans in the splitting case: high size workshops under medium uncertainty



a. (m=15,n=15,d=2)



a. (m=15,n=50,d=2)

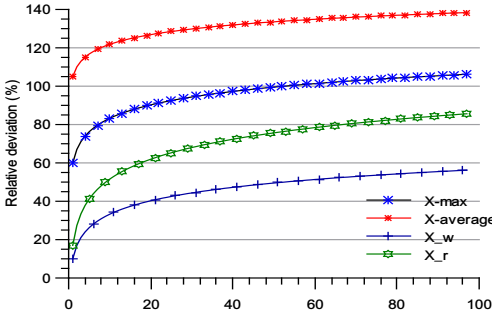


a. (m=15,n=200,d=2)

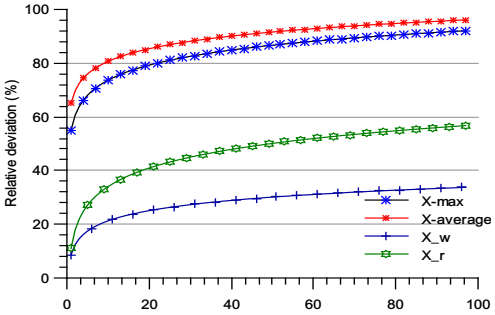
m=15	n	y	R^2
X_{max}	15	4.7ln(x)+29.06	0.977
	50	4.35ln(x)+19.34	0.973
	200	2.73ln(x)+16.43	0.961
$X_{average}$	10	4.9ln(x)+32.14	0.952
	50	4.58ln(x)+21.34	0.968
	200	2.8ln(x)+21.04	0.953
X_w	15	5ln(x)+9.1	0.974
	50	3ln(x)+7	0.957
	200	2ln(x)+4.3	0.966
X_r	15	4ln(x)+18	0.980
	50	3.5ln(x)+9	0.958
	200	2.2ln(x)+7.10	0.953

d. logarithmic trend line functions

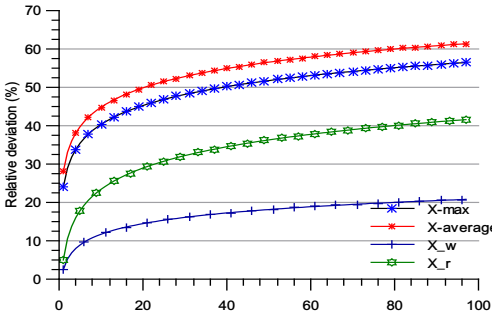
FIGURE E.16: Optimal robust solution worst-case makespans in the splitting case: high size workshops under high uncertainty



a. (m=15,n=15,d=3)



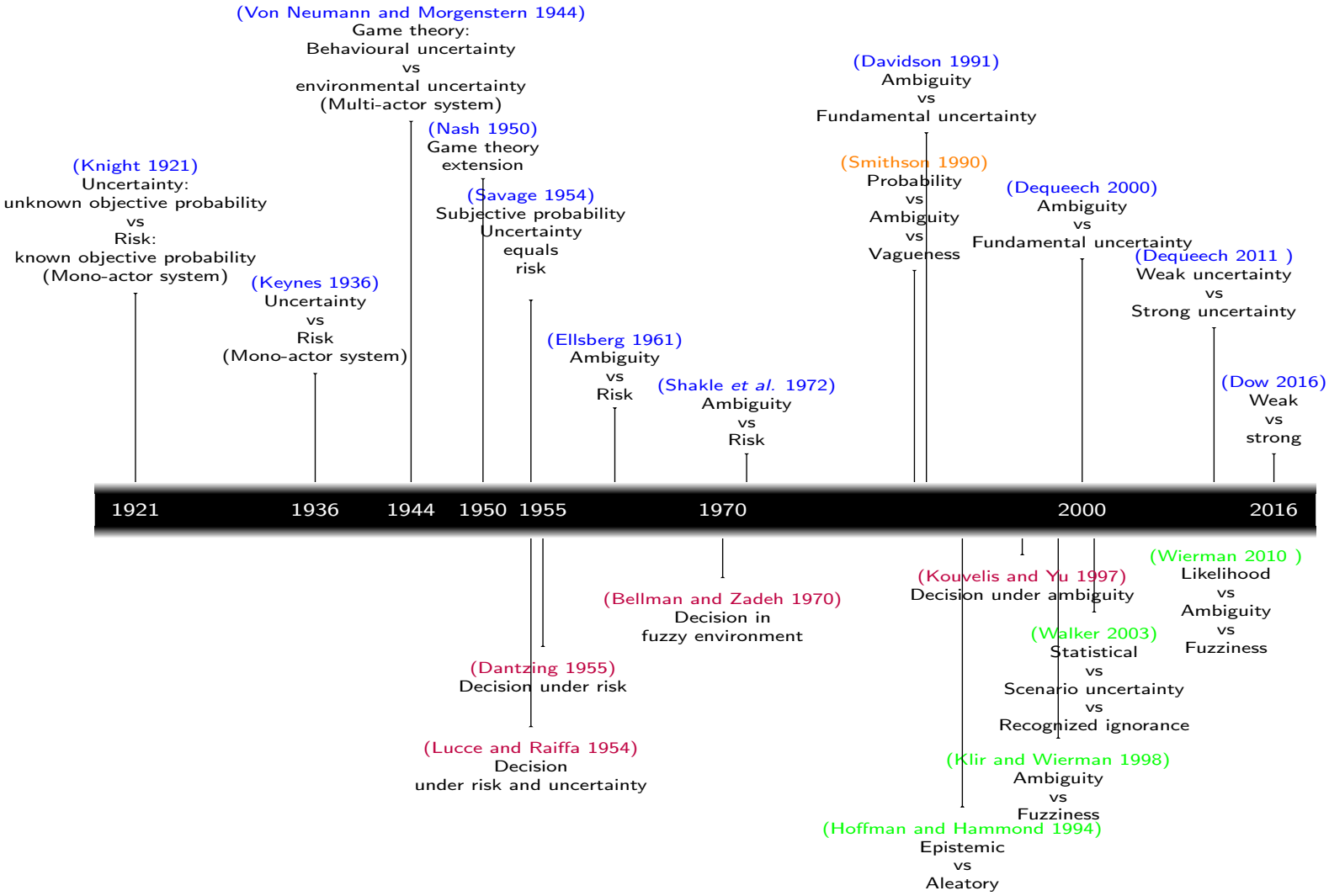
b. (m=15,n=50,d=3)



c. (m=15,n=200,d=3)

m=15	n	y	R^2
X_{max}	15	10.16ln(x) + 60.14	0.977
	50	8.10ln(x) + 55.04	0.973
	200	7.12ln(x) + 24.20	0.966
$X_{average}$	15	7.30ln(x) + 105.10	0.973
	50	6.82ln(x) + 65.10	0.978
	200	6.44ln(x) + 28.4	0.964
X_w	15	14.1ln(x) + 17	0.979
	50	5.5ln(x) + 8.6	0.970
	200	4ln(x) + 2.5	0.960
X_r	15	15ln(x) + 24	0.979
	50	10ln(x) + 11	0.970
	200	8ln(x) + 5	0.960

FIGURE E.17: Uncertainty concept timeline



Abstract: Scheduling on unrelated parallel machines is a common problem in many systems (as semi-conductors manufacturing, multiprocessor computer applications, textile industry, etc.). In this thesis, we consider two variants of this problem under uncertain processing times. In the first case, each job can be split into continuous sub-jobs and processed independently on the machines with allowed overlapping. In the second case which is termed preemption, we prohibit the overlapping. From a mathematical viewpoint, the splitting problem is a relaxed version of the preemptive problem. The objective is to minimize the makespan. The deterministic linear formulations provided by the literature allow solving these problems in polynomial times under the hypothesis of certainty. But when we consider uncertain processing times, these algorithms suffer from some limitations. Indeed, the solutions computed based on a nominal instance, supposed to be certain, turn usually to be suboptimal when applied to the actual realization of processing times.

Our approach consists in incorporating the uncertain processing times in these problems without making any assumption on their distribution. Hence, we use discrete scenarios to represent the uncertain processing times, and we adopt a proactive approach to provide robust solutions. We use special case policies that are commonly used in the industry to compute robust solutions. We show that the solutions based on some of those policies are potentially good in terms of robustness according to the worst-case makespan, especially the scenario s_{max} solution under which all the processing times are set to their maximal values. However, the robustness costs of these solutions are not satisfying. Thus, we propose to compute optimal robust solutions. For this purpose, we use a variable change that allows us to formulate and solve, in polynomial times, the robust versions of the considered scheduling problems. Moreover, the computational results assert that the robustness cost of the robust optimal solutions is not usually very high. Moreover, we assess the stability of the robust solutions under a new scenario induced by variations. In fact, the decision-maker is only responsible for the consequences of the decisions when the processing time realizations are within the represented uncertainty set. Thus, we define stability of a robust solution as its ability to cover a new scenario with minor deviations regarding its structure and its performance.

Our main contributions in this thesis are to provide a guide to understand uncertainty issues and their handling, in particular in scheduling on parallel machines. In this context, we have proposed a solving approach to help decision-makers compute robust solutions and choose among these solutions those with the most stable structure and the most stable performance.

Keywords: *robustness, min-max, min-max-regret, discrete processing time scenarios, parallel machines, stability analysis*