Scheduling optimization applied to the electricity, oil and gas industries

Mario Costa Levorato Junior

AVIGNON UNIVERSITÉ

THESIS

presented at the Avignon Université to obtain
the degree of Doctor

Graduate School Nº 536
Agrosciences & Sciences
Sciences, Technologies, Santé
Speciality : *Computer Science*

By

Mario COSTA LEVORATO JUNIOR

Scheduling optimization applied to the electricity, oil and gas industries

Optimisation de l'ordonnancement appliquée à l'industrie du pétrole

Research Unity : EA 4128 LIA – Laboratoire Informatique d'Avignon

Publicly supported on May 13, 2022 before a jury composed of :

Rapporteurs :	M. Agostinho Agra	Associate professor, Dep. de Matemática - Univ. de Aveiro, Portugal
(Reviewers)	Mme Sophie Demassey	Associate professor, CMA - Mines ParisTech, France
Examinateurs :	Mme Simone de Lima Martins	Associate professor, IC - Universidade Federal Fluminense, Brasil
(Examiners)	M. Igor Machado Coelho	Associate professor, IC - Universidade Federal Fluminense, Brasil
	Mme Ayse Nur Arslan	Assistant professor, IRMAR - INSA Rennes, France
	M. David Sotelo Pinheiro da Silva	Systems Analyst, Operations Research, Petrobras, Brasil
Encadrants :	M. Yuri Abitbol de Menezes Frota	Associate professor, IC - Universidade Federal Fluminense, Brasil
(Advisors)	Mme Rosa Figueiredo	Associate professor, LIA - Avignon Université, France
Directeur de thèse :	M. Serigne Gueye	Associate professor, LIA - Avignon Université, France

LABORATOIRE INFORMATIQUE D'AVIGNON

Titre : Optimisation de l'ordonnancement appliquée à l'industrie du pétrole

Mot clés : optimisation robuste, micro-réseaux, engagements flexibles, planification énergétique, ordonnancement, flow shop.

Résumé : Ce travail de recherche propose de nouvelles approches de solutions pour deux problèmes importants dans les domaines de l'énergie, du pétrole et du gaz, qui impliquent des paramètres d'entrée incertains.

Dans le contexte des réseaux d'énergie intelligents, le premier problème concerne la réalité des micro-réseaux qui échangent de l'énergie avec le réseau principal pour vendre leur surplus de production (provenant de sources d'énergie renouvelables) ou acheter un montant supplémentaire pour soutenir la demande des consommateurs locaux. Dans ce scénario, les dispositifs de contrôle intelligents sont des éléments importants, exécutant la planification énergétique en temps réel en fonction des fluctuations de la production et de la consommation. Comme on pouvait s'y attendre, la production et l'approvisionnement en électricité du réseau principal deviennent plus imprévus et risqués à mesure que les quantités d'énergie échangées oscillent au fil du temps. La première partie de la thèse étudie un framework de souscription des contrats énergétiques flexibles et bilatéraux, établis entre des fournisseurs d'électricité et un client. La souscription est liée à une stratégie de commande en temps réel (RTCS), adaptée à la planification énergétique des micro-réseaux avec des incertitudes dans la production et la consommation d'électricité.

Les principaux produits développés sont un modèle d'optimisation robuste multipériode embarqué, capable de fournir des solutions pour l'échange d'énergie sur plusieurs périodes, minimisant le coût par le client dans le pire des cas, ainsi qu'un ensemble de stratégies de contrôle pour la planification énergétique en temps réel. Au cours de la recherche, le modèle d'optimisation robuste initial a été amélioré pour représenter l'incertitude budgétisée, permettant des solutions moins conservatrices qui sont, en même temps, plus flexibles et moins chères, tout en assurant une protection contre les pires scénarios. La solution proposée a été testée à l'aide de données de consommation et de production collectées auprès d'un micro-réseau énergétique réel dans un centre de recherche à Tsukuba, au Japon.

S'appuyant sur un ensemble de contrats d'achat d'énergie inspirés du monde réel, les résultats de simulation ont confirmé l'efficacité de différentes stratégies de contrôle robustes, selon les types de scénarios d'incertitude. Pour des niveaux de protection spécifiques, les stratégies de contrôle robustes ont dominé les stratégies déterministes (naïves) dans toutes les mesures de coût opérationnel et de fiabilité du système. Les résultats obtenus avec une étude de cas montrent que l'efficacité de chaque solution robuste dépendra du profil de charge du micro-réseau et de la production renouvelable, qui varient selon la saison de l'année. D'où l'importance de l'ensemble d'incertitudes budgétées, qui fournit un ensemble de solutions robustes, avec différents niveaux de protection, parmi lesquelles le décideur peut choisir.

Le deuxième front de recherche est lié à la planification de la production sous incertitude, en particulier le problème d'ordonnancement connu sous le nom de *Robust Permutation Flow Shop Scheduling*. Nous utilisons l'approche de l'incertitude budgétaire, où les temps de traitement des opérations varient dans un intervalle donné. Le scénario le plus défavorable est borné par un paramètre de budget Γ, qui limite le nombre maximum d'opérations dont les temps de traitement peuvent osciller à leurs valeurs les plus défavorables. Le grand avantage de cette variante du problème consiste à ajuster le niveau de conservatisme de la solution, obtenant ainsi un équilibre entre le coût de la solution et la robustesse dans le pire des cas. Nous avons développé des méthodes de résolution pour deux fonctions objectifs différentes : *makespan* et *somme pondérée des temps d'exécution des tâches*. A notre connaissance, il s'agit du premier travail permettant d'obtenir des solutions robustes optimales aux deux objectifs.

Concernant la fonction objectif *makespan*, nous avons étendu deux formulations MILP classiques pour le cas déterministe et les avons combinées avec un cadre de génération de colonnes et de contraintes (*Column-and-Constraint Generation* - C&CG). A cet effet, un algorithme de programmation dynamique a également été développé, permettant l'identification des pires scé-

narios en temps polynomial. De nombreux résultats expérimentaux ont démontré que l'algorithme proposé était efficace pour obtenir des ordonnancements robustes optimaux pour des problèmes de petite et moyenne taille (y compris des instances de 50×2, 100×2 et 10×5, 15×5). De plus, sur la base d'une étude de cas avec deux instances représentatives, nous avons évalué le compromis entre la qualité de la solution et le coût, en comparant des solutions robustes à des solutions déterministes et stochastiques. De plus, selon des simulations basées sur trois distributions de probabilités, ces calendriers robustes ne présentaient qu'un faible surcoût dans le coût de solution attendu.

Nous avons également développé une métaheuristique de recherche adaptative randomisée gloutonne (GRASP) pour obtenir des solutions efficaces pour les instances de problèmes volumineux (jusqu'à 100×50). L'évaluation des performances de GRASP a nécessité l'adaptation d'instances de la littérature, telles que les instances bien connues de Taillard. Les résultats expérimentaux ont démontré que l'algorithme GRASP est efficace pour obtenir des ordonnancements robustes optimaux pour des problèmes de petite et moyenne taille, par rapport à la méthode de résolution exacte C&CG. L'évaluation était basée sur 4 ensembles de problèmes de test et il a été démontré que GRASP produisait des solutions optimales ou quasi optimales sur toutes ces instances.

Enfin, nous avons exploré le *flow shop* robuste avec l'objectif *somme pondérée des temps de réalisation des tâches* (*weighted sum of job completion times*, en anglais), où les temps de traitement des opérations sont sujets à l'incertitude. Dans le contexte de l'industrie pétrolière et gazière, cette variante du *flow shop* est associée au planning de maintenance des plates-formes pétrolières. Lorsque chaque équipement fait l'objet d'une maintenance, certaines opérations d'assistance doivent voir leur ordre d'exécution respecté. Et pour terminer cet ensemble défini de tâches, les puits de pétrole doivent être fermés uniquement pour rouvrir à la production à la fin du calendrier. L'objectif, dans ce cas, est de trouver un calendrier qui minimise la perte de production de pétrole causée par le temps pendant lequel chaque puits de pétrole est resté fermé pour maintenance. Sur la base du cadre de génération de colonnes et de contraintes, nous avons pu obtenir des solutions exactes pour des instances d'une taille allant jusqu'à 15×5. En plus, nous avons proposé une étude de cas appliquée à l'industrie pétrolière et gazière, en utilisant des données réelles, obtenues à partir de l'historique des plates-formes pétrolières brésiliennes.

Title: Scheduling optimization applied to the electricity, oil and gas industries

Keywords: Robust Optimization. Smart grid. Energy. Scheduling. Permutation Flow Shop.

Abstract: This thesis proposes new solution approaches for two important problems in the areas of energy, oil and gas, which involve uncertain input parameters.

In the context of smart grids, the first problem addresses the reality of microgrids which trade energy with the main grid to either sell its production surplus (from renewable energy sources) or buy an additional amount to support local consumers' demand. In this scenario, smart control devices are important elements, executing real-time energy scheduling according to fluctuations in production and consumption. As we might expect, the main grid's power generation and supply becomes more unscheduled and risky as energy trading quantities oscillate over time. The first part of the work studies a flexible bilateral energy contract subscription framework, established between electricity suppliers and a client. The framework is coupled with a real-time command strategy (RTCS), suited for energy scheduling of microgrids with uncertainty in both production and consumption.

The main products are an embedded Robust Optimization model, capable of providing solutions for multi-period-ahead trading of energy, while minimizing the microgrid's worst-case cost, as well as a set of control strategies for real-time energy scheduling. During the research, the initial robust optimization model was improved to represent budgeted uncertainty, allowing for less conservative solutions that are, at the same time, more flexible and less expensive, while providing protection against worst-case scenarios. The proposed solution was tested using consumption and production data collected from a real microgrid in a research lab in Tsukuba, Japan.

Relying on a set of real-world-inspired energy purchase contracts, simulation experiments have confirmed the efficacy of different robust-based RTCS strategies, according to scenario types. For specific protection levels, the robust RTCS was able to dominate the naïve deterministic RTCS in all operational cost and system reliability metrics. Results obtained with a case study show that the effectiveness of each robust solution will depend on the microgrid's load profile and renewable production, which vary according to the season of the year. Hence the importance of the budgeted uncertainty set, which provides a pool of robust solutions, with different protection levels, the decision-maker can choose from.

The second research avenue is related to production planning under uncertainty, in particular the scheduling problem known as *Robust Permutation Flow Shop Scheduling*. We adopt the budgeted uncertainty approach, where operation processing times are expected to vary within a given interval. Moreover the worst-case scenario is bounded by a budget parameter Γ, which limits the maximum number of operations whose processing times may oscillate to their worst-case values. The great advantage of this variant of the problem consists in adjusting the level of conservatism of the solution, thus obtaining a balance between solution cost and robustness in the worst case. We developed solution methods for two different objective functions: *makespan* and *weighted sum of job completion times*. To our knowledge, this is the first work to obtain optimal robust solutions to both objectives.

Regarding the *makespan* objective function, we extended two classical MILP formulations for the deterministic case and combined them with a Column-and-Constraint Generation (C&CG) framework. For this purpose, a dynamic programming algorithm was also developed, allowing the identification of worst-case scenarios in polynomial time. Extensive experimental results demonstrated that the proposed algorithm was effective in obtaining optimal robust schedules for small and medium-sized problems (including 50×2, 100×2 and 10×5, 15×5 instances). Additionally, based on a case study with two representative instances, we have assessed the trade-off between solution quality and cost, comparing robust solutions to deterministic and stochastic ones. Also, according to simulations based on three probability distributions, such robust schedules presented only a small overhead in the expected solution cost.

We also developed a greedy randomized adaptive search (GRASP) metaheuristic to obtain efficient solutions for large problem instances (up to 100×50). The evaluation of GRASP performance required the adaptation of literature

instances, such as the well-known Taillard instances. Experimental results have demonstrated that the GRASP algorithm is efficient in obtaining optimal robust schedules for small and medium-sized problems, when compared to the C&CG exact solution method. The evaluation was based on 4 sets of test problems and GRASP has been shown to produce optimal or near-optimal solutions on all of these instances.

Finally, we explored the robust flow shop with the *weighted sum of job completion times* objective, where the processing times of operations are subject to uncertainty. In the context of the oil and gas industry, this variant of the *flow shop* is associated with the maintenance schedule for oil rigs. When each piece of equipment is subject to maintenance, certain assistance operations must have their order of execution respected. Additionally, to complete this defined set of tasks, oil wells must be shut down only to reopen to production at the end of the schedule. The objective, in this case, is to find a schedule that minimizes the loss of oil production caused by the time that each oil well has remained closed for maintenance. Based on the Column-and-Constraint Generation framework, we were able to obtain exact solutions to instances of size up to 15×5. In addition, we proposed a case study applied to the oil and gas industry, using real data, obtained from the history of Brazilian oil platforms.

Título: Otimização de agendamento aplicada às indústrias de eletricidade, petróleo e gás

Palavras-chave: Otimização Robusta. Smart grid. Energia. Agendamento. Programação. Permutation Flow Shop.

Resumo: Esta tese propõe novas abordagens de solução para dois problemas importantes nas áreas de energia, petróleo e gás, que envolvem parâmetros de entrada incertos.

No contexto das redes elétricas inteligentes (*smart grids*), o primeiro problema aborda a realidade das micro-redes (*microgrids*) que comercializam energia com a rede principal para vender seu excedente de produção (oriundo de fontes renováveis de energia) ou comprar um volume adicional para atender à demanda dos consumidores locais. Nesse cenário, dispositivos de controle inteligentes são elementos importantes, executando a programação de energia em tempo real de acordo com as oscilações de produção e consumo. Como poderíamos esperar, a geração e o fornecimento de energia da rede principal apresentam mais incerteza e risco, pois as quantidades de energia comercializadas oscilam ao longo do tempo. A primeira parte do trabalho estuda uma estrutura flexível de assinatura de contratos bilaterais de energia, estabelecida entre fornecedores de energia elétrica e um cliente. A estrutura de contratação é acoplada a uma estratégia de comando em tempo real (RTCS), adaptada à programação de energia de microredes com incerteza tanto na produção quanto no consumo de eletricidade.

Os principais produtos são um modelo de Otimização Robusta multi-período embarcado, capaz de fornecer soluções para comercialização de energia em vários períodos, minimizando o custo pago pelo cliente no pior caso, bem como um conjunto de estratégias de controle para programação de energia em tempo real. Durante a pesquisa, o modelo inicial de Otimização Robusta foi aprimorado para representar a incerteza orçamentária (*budgeted uncertainty*), permitindo soluções menos conservadoras, mas ao mesmo tempo mais flexíveis e menos dispendiosas, sem abrir mão da proteção contra os piores cenários. A solução proposta foi testada usando dados de consumo e produção coletados de uma microrede instalada em um laboratório de pesquisa em Tsukuba, no Japão.

Baseando-se em um conjunto de contratos de compra de energia inspirados no mundo real, experimentos de simulação confirmaram a eficácia de diferentes estratégias de controle robustas, de acordo com os tipos de cenário de incerteza. Para níveis de proteção específicos, as estratégias de controle robustas foram capazes de dominar as estratégias determinísticas (ingênuas) em todas as métricas de custo operacional e confiabilidade do sistema. Os resultados obtidos com um estudo de caso mostram que a eficácia de cada solução robusta dependerá do perfil de carga da micro-rede e da produção renovável, que variam de acordo com a estação do ano. Daí a importância da utilização da incerteza orçamentária, que fornece um conjunto de soluções robustas, com diferentes níveis de proteção, entre as quais o decisor pode escolher a mais apropriada.

O segundo estudo envolve o planejamento da produção sob incerteza, em particular o problema de programação conhecido como *Robust Permutation Flow Shop Scheduling*. Adotamos a abordagem de Otimização Robusta conhecida como *budgeted uncertainty*, onde se espera que os tempos de processamento das operações variem dentro de um determinado intervalo. Além disso, o cenário de pior caso é limitado por um parâmetro de orçamento Γ, que limita o número máximo de operações cujos tempos de processamento podem oscilar para seus valores de pior caso. A grande vantagem desta variante do problema consiste em ajustar o nível de conservadorismo da solução, obtendo assim um equilíbrio entre custo da solução e robustez no pior caso. Métodos de solução foram então desenvolvidos para duas funções objetivo diferentes: *makespan* e *soma ponderada dos tempos de término dos jobs*. Até onde sabemos, este é o primeiro trabalho que conseguiu obter soluções ótimas robustas para ambas as funções objetivo.

Em relação à função objetivo *makespan*, estendemos duas formulações clássicas de MILP para o caso determinístico e as combinamos com uma estrutura de geração de colunas e restrições (*Column-and-Constraint Generation* - C&CG). Para tanto, também foi desenvolvido um algoritmo de programação dinâmica, que permite a identificação dos cenários de pior caso

em tempo polinomial. Extensos resultados experimentais demonstraram que o algoritmo proposto foi eficaz na obtenção de soluções robustas ótimas para instâncias de pequeno e médio porte (incluindo 50×2, 100×2 e 10×5, 15×5). Adicionalmente, com base em um estudo de caso com duas instâncias representativas, avaliamos o *trade-off* entre qualidade e custo da solução, comparando soluções robustas com determinísticas e estocásticas. Além disso, de acordo com simulações baseadas em três distribuições de probabilidade, os cronogramas robustos obtidos apresentaram apenas uma pequena elevação no custo esperado da solução.

Também foi desenvolvida uma metaheurística de busca adaptativa randomizada gulosa (GRASP) para obter soluções eficientes para instâncias grandes do problema (até 100×50). A avaliação de desempenho do GRASP exigiu a adaptação de instâncias da literatura, como as conhecidas instâncias Taillard. Resultados experimentais demonstraram que o algoritmo GRASP foi eficiente na obtenção de soluções ótimas robustas para instâncias de pequeno e médio porte, quando comparado ao método exato de solução C&CG. A avaliação foi baseada em 4 conjuntos de problemas de teste e o GRASP foi capaz de obter soluções ótimas ou quase ótimas em todas essas instâncias.

Por fim, exploramos o *flow shop* robusto com o objetivo *soma ponderada dos tempos de término dos jobs*, onde os tempos de processamento das operações estão sujeitos à incerteza. No contexto da indústria de óleo e gás, esta variante do *flow shop* está associada ao cronograma de manutenção das plataformas de petróleo. Quando cada equipamento é submetido à manutenção, determinadas operações devem ter sua ordem de execução respeitada. Além disso, para concluir esse conjunto definido de tarefas, os poços de petróleo devem ser fechados, podendo ser reabertos apenas no final do cronograma. O objetivo, neste caso, é encontrar um cronograma que minimize a perda de produção de petróleo causada pelo tempo que cada poço de petróleo permaneceu fechado para manutenção. Com base no *framework* de Column-and-Constraint Generation, conseguimos obter soluções exatas para instâncias de tamanho até 15×5. Além disso, propusemos um estudo de caso aplicado à indústria de óleo e gás, utilizando dados reais, obtidos a partir do histórico das plataformas de petróleo brasileiras.

TABLE OF CONTENTS

Chapter 1

Introduction

1.1 Context

This research proposes new robust solution approaches for two crucial problems in the energy, oil, and gas industries, involving uncertain input parameters. The first one concerns the operation of microgrids with uncertain production and consumption, including electricity trading and real-time energy scheduling. The robustness of the energy grid, i.e., the capacity to be resilient to demand fluctuations, has always been a topic of extreme importance. More recently, with the advent of hybrid energy systems, new issues related to renewable energy production (e.g., wind, biomass, sun) and consumption (i.e., electric cars, batteries) have become apparent. Optimization problems appear in the operation of these systems, and uncertainty ought to be considered.

The second problem concerns the Oil and Gas industry, particularly oil-well maintenance scheduling. Among several exploration and production problems, planning and scheduling of oil platforms are of main interest. A frequent objective in these problems is to determine the optimum values of decision variables to maximize profits, increase production rates and reduce production costs. Nonetheless, considering uncertainty in such optimization problems is extremely important, given the risky nature of the industry.

It is worth noting that the classic paradigm for decision making in Mathematical Programming consists in developing a model which assumes that the input data is accurate and equal to nominal values. In other words, the influence of data uncertainties on the quality and feasibility of the model are not considered. Therefore, if data can assume different values than the nominal ones, several restrictions of the problem under analysis can be violated. The optimal solution obtained from nominal data may not be appropriate or even possible. This context evidences the need to develop “robust” models, as flexible as possible to data uncertainty.

When optimal decisions are calculated based on uncertain data, naïve solution approaches include applying a margin of error to the decision or considering a set of cases to obtain a balanced judgment. Although useful, the success of these approaches depends on decision makers’ abilities and the completeness of available data. Moreover, if business complexity increases, it may be impossible to adopt these strategies. A common solution approach that has been used in Operational Research for a long time is Stochastic Optimization (SO) [53], whose fundamental premise states that the uncertainty can be described by using probabilities.

On the other hand, Robust Optimization (RO) [14] provides a different approach for optimizing problems under uncertain conditions. According to it, the objective and constraint functions are only assumed to belong to certain sets in function space. Uncertain problem parameters are modeled as belonging to a set of uncertainties, which contrasts with the more traditional modeling approach which uses probability distributions. The intention is to make a feasible decision no matter what the constraints turn out to be, and optimal for the worst-case objective function.

RO is a suitable modeling approach when the uncertainty interval is known, but not necessarily the probability distribution. Input data will assume an uncertain value in a specified fixed range. This way, calculated solutions will be available, considering all constraints, when model parameters vary within the ranges of uncertainty. If this is too restrictive, it is also possible to define with which probability the solution will satisfy restrictions. The robustness of decisions is measured in terms of better performance considering all input values.

While Stochastic Optimization can originate large models if several scenarios have to be analyzed (thus making it important to limit the number of scenarios considered, and unfortunately making the results less robust), Robust Optimization models in general grow only slightly when uncertainty is added and, therefore, can be solved efficiently.

RO is currently becoming a popular approach, applied in several areas of Mathematical Programming, ranging from Mixed Integer Linear Programming (MILP) to non-linear optimization. Moreover, its solution methods have already been applied to several problems in the energy, and oil and gas industries, including:

— Refinery operational planning [81] and scheduling of crude oil operations, an important part of overall refinery operations [143, 88].

— In Field Appraisal and Development, optimization of oil production under uncertainty (including the Recovery Factor (RF) of an oil field) can benefit from Robust Optimization [35, 158]. In particular, many RO models are focused on the recovery phases of a petroleum reservoir [35, 156, 158].

— Maritime transportation [1] is a common problem in the oil and gas industry, where routing problems are known to include many types of uncertainty related to unforeseen events (e.g., bad weather, mechanical breakdowns, and port congestion).

— Planning the transition to hybrid electric vehicules [51].

— Sustainable energy planning [7, 73, 41].

1.2 A robust contract collaboration framework for smart grids

In the context of smart grids, the first problem studied in this thesis is related to the collaboration established between a partner (energy suppliers) and its clients (individuals, households or

businesses, also known as end consumers) through the engagement on one or more day-ahead energy contracts. For a client's microgrid, the optimization framework allows for the two-way flow of electricity, with periodic energy market transactions (buying or selling energy), alternative energy sources (e.g., solar panels, wind turbines), electric cars, and energy storage systems (batteries). To our knowledge, this work is the first one to consider multiple energy contracts in microgrid energy management.

Undoubtedly, the proposed model's most significant feature involves mitigating the effects of uncertainty in energy production and consumption through Robust Optimization techniques. The final product is a real-time control algorithm embedded in smart control devices, providing a cost-effective strategy for buying and selling energy at each period of the day. Such strategy benefits not only the clients, which can make considerable savings, but also the partner, since the amount of energy bought out of engaged contracts tends to be minimized, thus reducing unexpected peaks in energy consumption (increasing predictability in consumer demands).

1.3 Robust Scheduling applied to Oil and Gas industry

Exploration and production of hydrocarbons [1] constitute a high-risk endeavor. While geologists deal with uncertainties related to structure, reservoir seal, and hydrocarbon charge, on the other side, economic evaluations also have to consider risk when analyzing costs, the economic viability of reservoirs, technology, and oil price. Indeed, even at the production stage, engineering parameters include a high level of uncertainties concerning critical variables, such as infrastructure, production schedule, quality of oil, and operational costs. All these uncertainties involve high-risk decision scenarios, which present no guarantee that hydrocarbon resources will be discovered and developed.

In fact, the need for decision-making under conditions of risk and uncertainty has always been common in the oil industry. However, only in 1956 the economics and risk of exploration were analyzed using probability theory and explicit modeling of oil and gas exploration stages [4]. Making important decisions in the petroleum industry requires the incorporation of significant uncertainties, long time horizons, multiple alternatives, and complex value issues into the decision model.

In this thesis, we focus on the maintenance scheduling of oil wells. When the pieces of equipment of each well go through maintenance, certain support operations must have their order of execution respected. The objective is to find a suitable sequence of maintenance tasks that minimizes the loss of oil production caused by the time each oil well remains closed for maintenance. The moment and the order in which the wells are closed and subsequently reopened impact the oil platform's production. Therefore, minimizing the time a set of wells stays closed for maintenance generates

1. Includes all the necessary tasks to obtain oil and gas products, such as basin and plan analysis, leads, prospect evaluation, development stages, facilities, logistics, and management.

financial gains (more oil will be produced), in the order of thousands and even millions of dollars.

In particular, we investigate a variation of the problem known as *Permutation Flow Shop*. Solving such scheduling problem taking real-world characteristics into account is a challenging task. First, there is the need to solve the problem quickly and efficiently. Additionally, in oil and gas companies, the data used as input to scheduling problems are actually forecasted, i.e., uncertainty is neglected in most cases. The solution methods proposed in this thesis will then focus on developing Robust Optimization models, and new solution approaches. Operation processing times are subject to uncertainty, and the lower and upper bounds of processing times are the only information available.

1.4 Challenges

As mentioned before, Robust Optimization provides a different approach to optimization problems under uncertain conditions. According to it, the unknown parameters are modeled as belonging to an uncertainty set, making it a more suitable modeling approach when the uncertainty set is known, but not necessarily the probability distribution. Unlike other robust optimization models, which generate only one conservative solution, the budgeted approach, applied in this thesis, allows the adjustment of the level of conservatism of the solution (protection level), enabling the incorporation of different attitudes toward risk (e.g., risk-averse, risk-neutral, or risk-seeking). As a result, the decision-maker can select the solution that achieves the best balance between robustness and optimality.

Besides the development of the solution method itself, when applying Robust Optimization, one major challenge concerns the quality evaluation of the obtained solutions, i.e., how it compares to deterministic (nominal) and stochastic solutions. Existing works from the literature have provided tools to accomplish this task. In particular, [16] proposed sensitivity analysis, which can be used to assess the price of increasing or decreasing the protection level. Measures such as the price of robustness and hedge value can be used to determine the trade-off between solution quality and cost, comparing robust solutions to deterministic ones.

Additionally, in the RO literature, it is well-known that such a hedge against worst-case costs might come at the price of higher expected costs in the long run. With this in mind, the Monte-Carlo simulation can be applied to compare robust, deterministic, and stochastic solutions, by obtaining a series of measures based on statistical data acquired from a vast number of simulation runs.

Even so, the proposal of comprehensive solution evaluation frameworks remains a challenge. In this thesis, we apply a mixture of the aforementioned techniques, which can be used to evaluate the quality of the obtained solutions and guide the decision-making process.

1.5 Contributions

In the first part of the thesis, concerning microgrid energy management, we propose the Contract Collaboration Problem (CCP) and its underlying multi-contract microgrid energy subscription framework, based on flexible commitments. We develop a Robust Optimization model under budgeted uncertainty to hedge against worst-case energy costs, coupled with Real-Time Command Strategies (RTCS) for microgrid energy trading and scheduling. An extensive case study is presented, including simulations on real microgrid data to evaluate the robust model performance. The results demonstrate the effectiveness of the proposed energy contract framework, which could be applied in the near future, along with the introduction of forward, short-term energy trading at the microgrid / local consumer level.

The second part of the thesis explores two robust scheduling problems derived from the classical Permutation Flow Shop Problem. We start with the robust version of the classical *makespan* objective, which is then used as a research path for the subsequent development of solution approaches to a second problem, based on the total weighted completion time objective.

Concerning the first scheduling problem, we propose an exact solution method, based on Column-and-Constraint Generation, and a GRASP metaheuristic to provide efficient solutions to problems of arbitrary size. In the solution process, we develop two robust counterpart formulations, along with a polynomial-time dynamic programming algorithm to enumerate the worst-case scenarios. The performance of the solution methods is evaluated, as well as the trade-off of robustness price and hedge value, according to different selections of the budget parameter value Γ. Solution quality assessment makes use of some techniques mentioned in Section 1.4.

We then solve the second robust flow shop problem, also under budgeted uncertainty, which now minimizes the worst-case total weighted completion time. The final intent is to provide cost-effective solutions to the real case of the oil-well maintenance scheduling problem, where the objective is to minimize the worst-case value associated with loss of oil production. Similarly to the previous problem, we use a Column-and-Constraint Generation algorithm to obtain exact solutions, based on seven proposed Robust Counterpart (RC) models. Good quality results and low average solution gaps are obtained on both synthetic and real-world problem instances, allowing the application of this solution method to day-to-day maintenance scheduling.

We propose a set of benchmark instances for both variations of the flow shop problem, along with a comprehensive case study to compare the obtained robust solutions to deterministic and stochastic ones, in terms of worst-case and expected solution values. According to simulations based on three probability distributions, it turns out that the cost overhead of the robust solutions is, in fact, negligible, i.e., when analyzing the empirical estimation of the expected solution cost of robust and stochastic solutions (as well as other statistical measures), there are no remarkable differences between them. Even though we cannot generalize it, the behavior observed in the case study does not

indicate “loss of quality” when adopting specific robust solutions instead of deterministic or stochastic ones.

In summary, the experimental results indicate the feasibility of applying robust solution methods to real-world problem instances, such as the ones from the oil and gas industry, whose current solutions are obtained through methods that disregard either uncertainty or the impact of worst-case scenarios. Based on their risk preferences, decision-makers can then choose an appropriate schedule from a pool of robust solutions with different levels of exposure to uncertainty.

Besides the obtained experimental results and observations about how robust solutions can be competitive in the long run, this thesis also contributes to data and code availability, since we publicly made available all datasets, source code, and results used throughout this thesis, as indicated in each chapter.

1.6 Personal Bibliography

All the contributions listed in Section 1.5 allowed us to make the following publications and submissions:

— **International Journals**

  — Mario Levorato, Rosa Figueiredo, Yuri Frota, Exact solutions for the two-machine robust flow shop with budgeted uncertainty, European Journal of Operational Research, 2021, Elsevier, ISSN 0377-2217, https://doi.org/10.1016/j.ejor.2021.10.021. [82]

  — Mario Levorato, Rosa Figueiredo, Yuri Frota. Robust microgrid energy trading and scheduling under budgeted uncertainty, Expert Systems with Applications, 2022, Elsevier, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2022.117471. [83]

— **International Conferences / Workshops**

  — Mario Levorato, Rosa Figueiredo, Yuri Frota, Antoine Jouglet, David Savourey. Real-time command strategies for smart grids based on the Robust Contract-based Collaboration Problem International Network Optimization Conference (INOC 2019), Jun 2019, Avignon, France [84]

— **National Conferences / Workshops with proceedings**

  — Mario Levorato, Rosa Figueiredo, Yuri Frota, Antoine Jouglet, David Savourey. Real-time energy scheduling for microgrids based on the Contract Collaboration Problem. 21eme Conferénce ROADEF de la Société Française de Recherche Opérationnelle et d’Aide à la Décision (ROADEF 2020), Feb 2020, Montpellier, France [85]

— Mario Levorato, Rosa Figueiredo, Yuri Frota, David Sotelo. The 2-machine robust flow shop problem under budgeted uncertainty. 22eme Conferénce ROADEF de la société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF 2021), Apr 2021, Mulhouse, France [86]

— **Submitted Articles for International Journals**

— Mario Levorato, David Sotelo, Rosa Figueiredo, Yuri Frota. Robust permutation flow shop total weighted completion time problem: solution and application to the oil and gas industry. Paper submitted to *Computers & Operations Research*, Elsevier, 2022.

1.7 Organization

The thesis comprises two different research problems. It is organized into four additional chapters, which we summarize as follows.

In the microgrid context, Chapter 2 presents the Robust Contract Collaboration Problem, the proposed robust framework and details an application to smart energy grids, including a realistic case study based on a research center microgrid in Japan.

Chapter 3 proposes exact and approximate solution approaches for the Robust Permutation Flow Shop Problem with the *makespan* objective, where operation processing times are uncertain and vary in a given interval. Based on the concept of budgeted uncertainty, the objective is to obtain a robust scheduling that minimizes the *makespan* of the restricted worst-case scenario, where only a subset of job processing times will oscillate to their worst-case values.

In its turn, Chapter 4 presents the Robust Permutation Flow Shop scheduling problem that minimizes the worst-case *total weighted completion time* objective under budgeted uncertainty. Experiments on real instances show that the obtained solution improvement is of great interest to the oil and gas industry.

Finally, Chapter 5 summarizes the research work and proposes some leads to solve the existing limitations. Some major perspectives are also identified.

Chapter 2

The Robust Contract Collaboration Problem (RCCP)

Given the advent of smart grids, we have witnessed several technological developments, and new electricity market rules and regulation mechanisms. The use of sensors, online connections, computational resources, and control strategies now plays a crucial role in ensuring reliability, reducing costs, and improving the efficiency of energy networks. Powered by renewable energy sources, microgrids can trade energy with the main grid to either sell its production surplus or buy an additional amount to support local consumers' demand, including flexible loads, such as smart appliances and electric vehicles. In this scenario, smart control devices are important elements, executing real-time energy scheduling according to fluctuations in production and consumption.

As we might expect, the main grid's power generation and supply become more unscheduled and risky as energy trading quantities oscillate over time. In this chapter, based on a flexible energy contract subscription framework, we propose real-time command strategies to be used by smart control devices, and suited for energy scheduling of microgrids with uncertainty in both production and consumption. The main contributions are a Robust Optimization model for contract subscription under budgeted uncertainty, and a set of heuristic control strategies for real-time energy scheduling. The robust model can provide solutions for multi-period-ahead trading of energy, while minimizing the restricted worst-case cost. An extensive computational case study, conducted on a real microgrid instance, has then confirmed the efficacy of the proposed solution approach.

2.1 Introduction to the problem

A microgrid consists of a small-scale integrated energy system that can manage its own generation and storage resources to dynamically supply local consumers' electricity demands [80]. Since a microgrid can integrate various sources of distributed generation, especially Renewable Energy Sources (RES), an increasing participation of microgrids can help relieve the supply tension of conventional generators in the main grid. However, the high fluctuation of RES production makes energy management more complex and uncertain. Consider, for example, a microgrid powered by a photo-voltaic system. Even if energy consumption follows a regular pattern, it is difficult to predict

its renewable generation accurately as it is subject to sudden weather changes. Consequently, the microgrid will present a volatile production profile and sometimes its energy storage capacity may not be able to cope with the instant demand for energy. Therefore the subsequent decision, how much electricity to buy, will inherit a considerable level of uncertainty, which is also undesirable for the main grid, since it introduces risk and higher operational costs.

Other relevant issues concerning microgrids have arised with the introduction of free energy markets. Consumers became able to produce energy (thus being called prosumers) and, in parallel, contract types, market models and pricing schemes have evolved [96, 65, 100, 5]. In liberalized markets, large-scale generators, suppliers, industrial consumers and other financial intermediaries trade energy in wholesale markets, including day-ahead auctions, where agents submit their bids and offers for delivery of electricity for each hour of the following day, before market closing time. On the other hand, small-scale prosumers are currently serviced by large suppliers in the retail market. Nonetheless, forward energy trading is expected to happen at the local level, with microgrids and actively managed distribution networks becoming more widespread [23]. As new challenges related to local purchasing fluctuations of prosumers arise, the main grid must regulate and stabilize the microgrids' energy purchasing behaviors. One way to accomplish this objective is to introduce flexible commitments contracts [133].

This study addresses a new framework for microgrid energy trading, with the novel introduction of flexible commitments in a multiple contract setting. The impacts of this contract-based framework are also investigated from the viewpoint of microgrid energy management. Consider a time horizon divided into discrete time periods. For each period, one or more contracts are offered, defining either selling or purchasing commitments, and providing the flexibility to trade energy between minimum and maximum amounts. Bearing in mind the uncertain nature of the renewable resources, the prosumer must choose the contracts for the whole time horizon, to minimize the worst-case cost, while guaranteeing that each commitment will be honored. In a second level of decision, in each time period, following the list of engaged contracts and minimum/maximum commitment constraints, a real-time scheduler coordinates the microgrid's systems, making energy trading and transfer operations, according to current storage units' status and instantaneous information regarding local electricity production and demand.

Although microgrid energy dispatch has been well studied in the literature, existing methods do not investigate the subscription to multiple and flexible electricity contracts, or even commit to future energy usage, according to forward markets. The same holds for works focused on dynamic electricity pricing [96, 65]. The closest work is by [32], which, based on Stochastic Optimization, proposed a dynamic contract mechanism to smooth out fluctuations of microgrids' purchasing from the main grid with time-specific commitments. Their research, however, assumes a single dynamic contract for the whole time horizon, in which the microgrid buys electricity from the energy company.

Among the benefits of flexible contracts, it enables small customers to engage in a set of short-

term contracts and spread energy purchasing decisions over a period of time. Besides avoiding the risk of relying on a single energy contract, the client also has the option to sell the contracted energy back to the grid and start over, which could be used to hedge against risk.

Flexible contract engagement may have drawbacks as well. Intelligent energy scheduling strategies are needed, a smart meter is essential to take accurate readings, and purchasing decisions are often more complex, with the client more exposed to risk, as market prices can go up or down. Moreover, if the client needs to buy out of any engaged contract, the energy price will be higher than the existing contract prices. For these reasons, the choice of energy contracts should be robust enough to protect the consumer even in the worst-case scenario, given its operational constraints. Traditional modeling approaches for handling uncertainty include Robust Optimization (RO) and Stochastic Optimization (SO). In this work, RO was chosen for two main reasons. First, probability distributions for energy production/consumption are generally unknown for recently-installed microgrid energy systems. Second, SO methods typically rely on scenario trees for modeling uncertainty, which makes them computationally expensive [102]. When applying a RO approach, the obtained models have improved tractability with less computational effort.

In a conference work [87], we introduced the first robust microgrid energy management model based on an electricity contract subscription framework with flexible commitments. In that previous work, we assumed a conservative box-shaped uncertainty set and obtained preliminary computational results on a single realistic case-study instance. In the present work, we extend both model and real-time command strategies: a budgeted uncertainty robust counterpart, which controls the level of solution conservatism, is described [16] and used in a look-ahead strategy for the real-time energy scheduling. We also present extensive computational experiments on a set of multiyear and seasonal case-studies based on data from a Japanese research center microgrid recently described in the literature [140].

The main features of this work are summarized as follows. Section 2.2 introduces the Contract Collaboration Problem and the underlying framework. Once the problem has been formally defined, Section 2.3 presents the state of the art on microgrid energy trading and management literature, and related works on Robust Optimization. In the context of the forward electricity market for microgrids, we describe in Section 2.4.2 a mathematical model for multi-contract energy trading with flexible commitments. A robust version of this model is then presented in Section 2.5, minimizing the costs and protecting against the worst-case realization of local production and consumption of electricity under budgeted uncertainty. Then, in Section 2.6, we describe real-time command strategies for energy scheduling within the microgrid, considering the contracts previously selected by our model. Finally, we present a case study, based on a real microgrid, detailing the results of these scheduling strategies, when coupled with the robust model solution (list of engaged contracts), in contrast with a deterministic approach to solving the problem.

2.2 The Contract Collaboration Framework

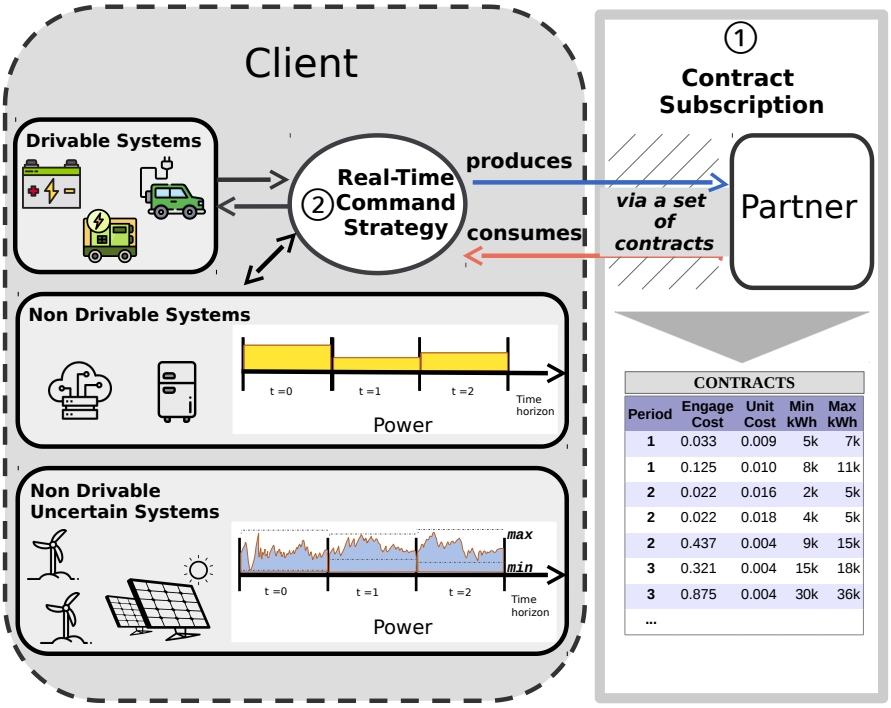

Figure 2.1 – The Contract Collaboration framework

Considering the context of demand response and smart grids [121], we propose a Contract Collaboration framework, established in two phases (Figure 2.1). It consists of an approach to handling energy management in microgrids, along with purchase/sell contracts based on flexible commitments.

The first level of decision concerns the list of contracts the client can subscribe to, at each time period, given the microgrid's energy demands and operational constraints (*Contracts* table in Figure 2.1). The solution to the so-called Contract Collaboration Problem (CCP), formally described in Section 2.2.1, provides the client with a commitment planning for the time horizon (i.e. which contracts to engage at each time period). Once this decision has been made, the list of engaged contracts cannot be changed for the whole time horizon.

In a second level of decision, inside each time period, a Real-time Command Strategy (RTCS) is responsible for performing on-line energy scheduling (item 2 in Figure 2.1). As explained in Section 2.2.2, the RTCS follows a predefined strategy to balance energy demand and supply at each instant of time, considering renewable energy sources, storage devices, drivable systems and deciding how much energy will be traded via each engaged contract.

2.2.1 The Contract Collaboration Problem (CCP)

The contract collaboration is established between two entities, both producers and consumers of a same kind of energy resource. One entity is called the $client$ (individuals, households or businesses) and the other one the $partner$ (energy supplier). These two entities have to collaborate in order to balance their consumption and production over a given time horizon. We consider a set $\{T_0, T_1, \ldots, T_{\bar{t}}\}$ of time points dividing the given time horizon into a set $\mathcal{I} = \{I_0, \ldots, I_{\bar{t}-1}\}$ of $\bar{t}$ time periods where $I_t = [T_t, T_{t+1})$, for each $t \in \{0, 1, 2, \ldots, \bar{t} - 1\}$.

The collaboration between the entities is established by the use of a set C of contracts of consumption or production, both offered by the partner. Let $C_t \subseteq C$ be the subset of contracts offered by the partner at time period I_t. Each contract $c \in C_t$ has its own functional constraints and its own gain/cost functions. The partner determines the set C_t and sets a price to engage each contract $c \in C_t$. On each time period $I_t \in \mathcal{I}$, the client is free to enter into a commitment with the partner through any subset of C_t. However, these commitments have to be taken by the client at the beginning of the time horizon and must be honored. At any time period, the $client$ also has the option to buy energy out of any engaged contract, but at a higher cost which can vary with the time period.

The client's microgrid is composed by a set of systems S that produce/consume energy, each one with its own functional constraints and a cost/gain of consuming/producing over the time periods. In particular, the consumption/production can be driven for a subset of these systems ($drivable\ systems$) while the consumption/production is already planned for the other ones ($non\text{-}drivable\ systems$). Drivable systems are devices that allow being turned on/off or that must be loaded/unloaded from time to time (e.g., batteries, electric car), whereas non-drivable systems must be permanently turned on. Additionally, some of the drivable systems can store the energy resource under a capacity constraint and provide it when it is needed, thus being called $storage\ systems$. The uncertainty considered in the problem lies in a subset of the $non\text{-}drivable\ systems$, for which only uncertain previsions of the consumption/production are known. The so-called $uncertain\ non\text{-}drivable\ systems$ include, for example, renewable generation and variable energy consumption.

The Contract Collaboration Problem (CCP) consists in determining a cost-minimizing contract subscription from the client to the partner that satisfies all client-side consumer demands over the time horizon, and also guarantees that each commitment taken by the client with the partner is honored. For a detailed description of the CCP models, we refer the reader to Sections 2.4 and 2.5.

2.2.2 The Real-Time Command Strategy (RTCS)

In the first level of decision, the list of engaged contracts (i.e., the solution of the CCP problem) is set. The second level of the framework is in charge of defining a Real-Time Command Strategy (RTCS) that guarantees these contracts will be honored. The RTCS operates on smart control devices, being in charge of making scheduling decisions according to instantaneous energy supply

and demand fluctuations observed on customers' premises. Thanks to the development of advanced metering and communication infrastructure, these control devices have the ability to regulate energy consumption by directly communicating to the energy supplier and to other devices in the microgrid so as to prevent system overloads. Interesting examples would be the load reduction of a set of electric vehicle charging stations and the automatic activation of a group of electric generators.

The RTCS consists of a heuristic strategy that schedules, in real-time, the set of actions to be taken in order to properly manage the client's microgrid. From a real-time point of view, inside each time period $I_t \in \mathcal{I}$, the instantaneous production/consumption of each microgrid system is measured every Δ time units. It is also at this time scale that drivable systems are driven, i.e, every Δ time units a scheduling decision has to be taken by the control device, embedded in the client's microgrid, considering its state. For example, according to energy load and in real time, a group of generators (a type of *drivable system*) may be switched on during a period of higher demand and, analogously, a set of *storage systems* such as batteries can store energy during off-peak times in order to ease high demand supply in peak periods.

Regarding the energy contracts, the RTCS is in charge of deciding how much energy will be bought or sold, given each engaged contract and its minimum and maximum commitments. For instance, given a time period, if the client engages contract c, energy quotas (for buying or selling electricity) are established for each time period and, analogously, for each time slot.

The main objective of the RTCS is to reduce power consumption costs and promote load balance, while dealing with the effects of uncertainty in both production and consumption of energy. As we can expect, the RTCS operates subject to the CCP constraints, guaranteeing both contract commitments and energy balance at each moment. A full description of the real-time command strategies developed in this work is available in Section 2.6.

2.3 Literature review

In this section, we highlight existing works involving the two subjects addressed in the paper: microgrid energy scheduling (RTCS) and electricity trading models (such as the CCP). Also, we review works presenting robust approaches to related problems.

2.3.1 Microgrid energy scheduling and electricity trading models

Various approaches have been proposed to optimize microgrid operational schedules, with distinct objectives, constraints, and methods for handling uncertainty. As a consequence, different terms have been used to refer to similar real-time control mechanisms: microgrid energy scheduling [106, 164, 131, 160], real-time scheduling [57], real-time control system [152, 153], real-time power management and control [25], energy management policies [104], energy dispatching poli-

cies [151], microgrid energy management [163, 139] and microgrid operation [59, 116]. Some authors have also studied the islanded-mode operation of microgrids [56, 60].

In this review, we will focus on grid-connected microgrids, since energy trading with the main grid is a main premise of our work. As far as grid-connected microgrids are concerned, many works have applied risk-averse optimization methods to energy scheduling, dealing with uncertainties in several model parameters: energy prices, solar power production, wind power generation, Plug-in Electric Vehicle (PEV) consumption and availability, and load demand. An extensive, but not exhaustive, list of papers on microgrid energy scheduling is presented in Table 2.1, including the approach used to deal with uncertain data (SO or RO), which microgrid elements are assumed to be uncertain and the type of contract with the main grid.

The common point of existing works is how the microgrid interacts with the external energy market: energy transactions are modeled through a single contract (often with the utility grid/retail market), via purchase and sale prices that may vary in time, sometimes including a minimum/maximum tradable energy amount. One exception is the work of [106], where the prosumer can have at most two active contracts: one with a retailer and one with the grid company. In other words, flexible contract frameworks are not investigated. Such type of contract allows buying/selling from/to the main grid not only at different prices (even in the same time period), but also from/to different energy companies at the same time.

Ref	SO	RO	Uncertainty	Solution Method	Contract
[25]	x		PV, spot price	Decentralized multi-agents	Single (Spot market)
[32]	x		REN, load demands	Stochastic Dynamic Programming	Single dynamic contract
[106]	x		Energy loads, energy prices	General stochastic MILP model	1 contract with Utility grid + 1 dynamic price contract
[151]	x		Wind, PEV	Scenario generation and reduction	Single (Utility grid)
[131]	x		Wind, PV	Two-stage stochastic model (with scenarios)	Single (Utility grid)
[58]	x		REN, energy prices, energy consumption	Constrained stochastic programming; Lyapunov optimization	Single (Utility grid)
[104]	x		REN, load demands	Stochastic dynamic programming	Single (Utility grid)
[152]	x		PEV, PV, home load demand	Stochastic Dynamic Programming	Single (Utility grid)
[99]	x		Wind, PEV, energy prices	Multi-objective ILP and scenario analysis	Single (Utility grid)
[160]	x		Residual load	Chance-constrained optimization	Day-ahead commitments
[139]	x		PV, Wind	Bilevel stochastic MIP	Multiple contracts, but can only subscribe to one
[153]	x		PEV	Stochastic Dynamic Programming	Single (Utility grid)
[163]		x	REN	Dual decomposition and distributed subgradient	Single (Spot market)
[164]		x	REN	Dual decomposition and distributed subgradient methods	Single (Spot market)
[144]		x	Net demand, heat demand, and electricity price	Chance constraint approximation and RO (budget-constrained & distribution uncertainty)	Single (Utility grid)
[59]		x	CHP, electrical loads	Budget-constrained min-max robust counterpart	Single (Utility grid)
[27]		x	Wind	Scenario-robust MILP based on realistic weather forecast scenarios	Single (Utility grid)
[57]		x	REN, load demand	Day-ahead scheduling with two-stage RO	Single (Utility grid)
[116]		x	PV	Two-stage RO with budget constraints	1 firm contract and 1 non-firm contract

Table 2.1 – Summary of the works listed in literature review. SO (Stochastic Opt model), RO (Robust Opt model). Uncertainty: list of uncertain parameters. REN (Renewable energy production), CHP (Combined Heat and Power), PEV (Plug-in Electric Vehicle), Wind (Wind Generator), PV (Photo-Voltaic). Contract: type of contract used to buy and sell energy.

The novelty in our work is the incorporation of a multi-contract subscription framework for microgrids, based on flexible commitments. Even though such contract model is not present in existing low-voltage energy markets, it can be applied as an extension to the forward market (via an aggregator or similar energy service provider), or in local microgrid markets, following market structures depicted in [105, 72].

To our knowledge, only two authors present research directions similar to ours. From the viewpoint of multiple energy contracts, in [139] different Generation Companies can offer buy/sell contracts to the microgrid. However, although the microgrid can receive contract offers from different competing companies, it can select at most one contract for the whole time horizon. As far as contract flexibility is concerned, [160] explored the stochastic scheduling of microgrids where energy exchange must be made with day-ahead commitments. In the proposed market structure, the microgrid may be either rewarded for respecting existing commitments, or penalized for deviating too much from them. Despite this flexibility, their work assumes a single long-term contract with the utility company.

2.3.2 Related works on Robust Optimization

The retailer-supplier problem is a NP-hard supply chain problem in which decisions are normally taken in a finite horizon of time periods. The retailer must order some products from a supplier for each period, being sometimes allowed to change these orders according to customer's demand. Ordering, holding and shortage costs are involved. Demand uncertainty is frequent in this scenario and should be taken into account. Flexible commitments [11] is a coordination mechanism that aims to assist both retailers and suppliers in handling the uncertainty in customer's demand in a cooperative way. In this model, the retailer-supplier relation is established by contracts through which the retailer may have some flexibility to purchase quantities that actually deviate from his original commitments. Some variants of this mechanism were proposed in the literature, including orders that can be changed along the time horizon, sometimes incurring in penalty costs. The retailer-supplier problem with flexible commitments (RSFC problem) is a NP-hard problem.

[12] applied the Affinely Adjustable Robust Counterpart methodology (AARC) to the RSFC problem with uncertain demand, based on the box uncertainty set. First, the retailer places his commitments for each time period. Then, at the beginning of each time period, the retailer is allowed to order a quantity not necessarily respecting his commitment, but subject to certain lower and upper bounds. Cumulative bounds are also imposed, and the supplier specifies a penalty that must be paid due to any deviation between committed and actual orders. The cost function involved is minimized against the worst-case demand occurrence (min-max criterion). In the present work, the applied solution approach presented in Section 2.5 is similar to the one used by [12].

Besides the RO works listed in Table 2.1, which directly involve microgrids, we also refer the reader to additional RO works on related problems with uncertain demand and production of elec-

tricity. The first one involves the application of constraint generation and duality-based reformulation to solve the robust multistage Unit Commitment Problem, using both budgeted uncertainty and a customized dynamic uncertainty set [91]. The second work [26] involves an aggregator of residential prosumers, which participates in the day-ahead energy market to minimize operation costs. Budgeted uncertainty is considered in energy prices, demand and PV production, and Adjustable Robust Optimization is employed. The model takes into account energy purchases in the wholesale market, with the possibility of buying additional blocks of energy. However, no flexible commitments were considered in this framework.

2.4 A deterministic version of the CCP

In this section, we will present a simpler, deterministic model version of the CCP, denoted as DCCP, where the value of the uncertain parameters are assumed to be known in advance. The formulation comprises each microgrid component, its operational constraints, as well as the underlying contract subscription framework. The main idea of the CCP model is to find a cost-minimizing solution which provides the client with a list of energy contracts to engage in each time period, considering the whole time horizon. In the rest of this text, time is discretized into periods as indicated by $I_0, .., I_{\bar{t}-1}$ and energy units are assumed to be in kWh. Moreover, we will often write t for a time period I_t.

2.4.1 Nomenclature

Sets

$\mathcal{I}$ Time periods: $\mathcal{I} = \{I_0, .., I_{\bar{t}-1}\}$
T Time period indexes: T=$\{0, .., \bar{t} - 1\}$
$C_t \subseteq C$ Contracts offered by partner at period I_t
$S_D \subseteq S$ Certain drivable systems
$S_{ND} \subseteq S$ Certain non-drivable systems
B (Certain, drivable) storage systems

Input parameters - Partner

$v_c^t \geq 0$ fixed cost paid by the client for engaging contract $c \in C_t$ at period I_t, $\forall\, t \in T$
$\alpha_c^t \geq 0$ cost per energy unit consumed/provided according to contract $c \in C_t$ at period I_t
$\Pi_{t,c}^-$ minimal energy quota for contract $c \in C_t$ during period I_t
$\Pi_{t,c}^+$ maximal energy quota for contract $c \in C_t$ during period I_t
$\beta^t \geq 0$ cost per energy unit consumed by the client at period I_t not provided by engaged contracts, $\forall\, t \in T$

Input parameters - Client

$\delta^t \geq 0$ length of period I_t: how many slots of Δ time units compose this time period
P_s^t energy consumption(< 0)/production(> 0) of $s \in (S_D \cup S_{ND})$ during the whole time period I_t
v_s cost per energy unit produced(> 0)/consumed(< 0) when using system $s \in (S_D \cup S_{ND})$
$P^{min}{}_s^t$ minimum energy to be produced(> 0)/consumed(< 0) by drivable system $s \in S_D$ before the end of period I_t

$v_s \geq 0$ cost of energy unit charged/discharged by storage system $s \in B$
$u_s^{min} \geq 0$ minimal storage level of system $s \in B$
$u_s^{max} \geq 0$ storage capacity of system $s \in B$
$u_s^0 \geq u_s^{min}$ initial storage level (at period I_0)
$0 \leq \lambda_s \leq 1$ the loss coefficient of system $s \in B$
$\theta_s^{abs} > 0$ maximum energy stored in $s \in B$ during Δ time units
$\theta_s^{ref} > 0$ rated capacity, i.e., maximum energy delivered during Δ time units

Model variables

$y_c^t \begin{cases} 1 & \text{if the client engages contract } c \in C_t \text{ at period } I_t \\ 0 & \text{otherwise} \end{cases}$

q_c^t amount of electricity sold (< 0) / bought (> 0) by the client at time period I_t related with contract $c \in C_t$
$0 \leq x_s^t \leq 1$ percentage of time period I_t drivable system $s \in S_D$ is used
$g_s^t \geq 0$ energy fed into storage system $s \in B$ during period I_t
$h_s^t \geq 0$ energy consumed from storage system $s \in B$ during period I_t
$r_s^t \geq 0$ amount of energy stored in $s \in B$ at time period $t \in T \cup \{\bar{t}\}$
$e^t \geq 0$ extra amount of energy requested by the client (out of any engaged contract) at time period I_t

Given a set of contracts offered by the partner, each contract $c \in C_t$ is associated with a time period I_t and its fixed (v_c^t) and variable prices (α_c^t) may vary if the period is in peak hours or off-peak. By engaging in a contract, the client must respect the established energy quotas $\Pi_{t,c}^-$ and $\Pi_{t,c}^+$, that may be positive (if the client purchases energy from the partner) or negative (the client sells energy to the partner). The partner can also sell energy to the client out of any engaged contract at a specific unit price β^t.

The information about energy consumption (or production) P_s^t of all *drivable* and *non-drivable systems* is discretized into time periods I_t. Moreover, in the microgrid's energy scheduling, each time period I_t is further subdivided into δ^t **time slots of size** Δ, where Δ is an input parameter. At this time scale, the instantaneous production/consumption of each microgrid system is measured.

Each system $s \in S_D \cup S_{ND} \cup B$ may have an associated operational cost v_s (e.g. energy produced by a fuel generator has positive cost). This cost can also be zero (e.g. consumer system such as a building). Additionally, for each drivable system $s \in S_D$, we define minimum requirements for energy production/consumption at time period I_t. In other words, for each consumer *drivable system* s, ${P^{min}}_s^t > 0$ means system s must be supplied with ${P^{min}}_s^t$ units of energy before the end of time period I_t (e.g. when charging an electric car). Normally, ${P^{min}}_s^t = 0$ if s is a producer *drivable system*.

Last but not least, the *storage systems* are a key component for the success of the contract subscription framework. A set of batteries can store energy during off-peak time periods in order to ease high demand supply in peak periods. Besides the unit cost, there are several battery-specific parameters related to the storage levels, capacity and efficiency: u_s^{min}, u_s^{max}, u_s^0, λ_s, θ_s^{abs} and θ_s^{ref}.

2.4.2 Mixed-Integer Linear Programming formulation

Based on the model variables defined, we present a Mixed-Integer Linear Programming (MILP) model, whose optimal solution provides the client with a commitment planning for the whole time horizon: which contracts to engage in each time period.

$$\text{Min} \sum_{t\in T}\sum_{c\in C_t}(v_c^t y_c^t + \alpha_c^t q_c^t) + \sum_{s\in S_D} v_s \sum_{t\in T} P_s^t x_s^t + \sum_{s\in B} v_s \sum_{t\in T}(g_s^t + h_s^t) + \sum_{t\in T}\beta^t e_t \tag{2.1}$$

$$\textstyle\sum_{t'\leq t} P_s^{t'} x_s^{t'} \geq P^{min}{}_s^t,\ \forall\, t\in T,\ \forall\, s\in S_D : P_s^t > 0 \tag{2.2}$$

$$\textstyle\sum_{t'\leq t} P_s^{t'} x_s^{t'} \leq P^{min}{}_s^t,\ \forall\, t\in T,\ \forall\, s\in S_D : P_s^t < 0 \tag{2.3}$$

$$h_s^t \leq r_s^t,\ \forall\, t\in T,\ \forall\, s\in B \tag{2.4}$$

$$r_s^0 = u_s^0,\ \forall\, s\in B \tag{2.5}$$

$$u_s^{min} \leq r_s^t \leq u_s^{max},\ \forall\, t\in T\cup\{\bar{t}\},\ \forall\, s\in B \tag{2.6}$$

$$r_s^{t+1} = r_s^t - h_s^t + \lambda_s g_s^t,\ \forall\, t\in T,\ \forall\, s\in B \tag{2.7}$$

$$g_s^t \leq \theta_s^{abs}\delta^t,\ \forall\, t\in T,\ \forall\, s\in B \tag{2.8}$$

$$h_s^t \leq \theta_s^{ref}\delta^t,\ \forall\, t\in T,\ \forall\, s\in B \tag{2.9}$$

$$\Pi_{t,c}^- y_c^t \leq q_c^t \leq \Pi_{t,c}^+ y_c^t,\ \forall\, t\in T,\ \forall\, c\in C_t : \Pi_{t,c}^+ > 0 \tag{2.10}$$

$$\Pi_{t,c}^+ y_c^t \leq q_c^t \leq \Pi_{t,c}^- y_c^t,\ \forall\, t\in T,\ \forall\, c\in C_t : \Pi_{t,c}^- < 0 \tag{2.11}$$

$$\textstyle\sum_{c\in C_t} q_c^t + \sum_{s\in S_{ND}} P_s^t + \sum_{s\in S_D} x_s^t P_s^t + \sum_{s\in B}\lambda_s h_s^t + e^t \geq \sum_{s\in B} g_s^t,\ \forall\, t\in T \tag{2.12}$$

The objective function (2.1) includes, respectively: (i) the fixed costs involved in the client-partner engagement through a set of contracts; (ii) the sum of costs/gains of consuming/providing the amounts of electricity predicted by the set of engaged contracts; (iii) the costs of using drivable systems; (iv) the costs of using storage systems (including depreciation); (v) the costs of consuming extra amounts of electricity not predicted in the set of engaged contracts.

There are also costs associated with the use of non-drivable systems. However, since these costs are fixed, they do not need to be included in the objective function to be minimized.

Constraints (2.2)-(2.3) ensure the minimum usage of *drivable system* s, in case it produces (2.2) or consumes (2.3) electricity. Constraints (2.4) restrict the amount of electricity consumed from *storage system* s during a time period to be at most the amount stored. Additionally, constraints (2.5)-(2.6) state that the initial, minimum and maximum capacities of *storage system* s must be respected. Constraints (2.7) determine the amount of electricity stored on *storage system* s at the next time period. It must take into account its loss coefficient λ_s, i.e., when storing g_s^t kWh of energy, only λ_s % is effectively stored in s. Remark that h_s^t includes the amount of energy provided by s as well as the energy lost during this operation. The maximum quantity of energy that can be stored by *storage system* s during a time period t is guaranteed by constraints (2.8), while constraints (2.9) ensure the maximum quantity of energy that can be provided by a *storage system* s during a time period t.

Constraints (2.10)-(2.11) establish minimum and maximum quotas for contracts. They also guar-

antee that a non-zero consumption/production related with a contract available at a certain time period will imply an engagement to it.

Finally, constraints (2.12) define the electricity balance at each time period. When calculating the energy refunded by storage systems s, these inequalities must take into account the amount of energy lost during discharge, therefore h_s^t must be reduced proportionally to λ_s %. Besides, at any time period, total consumption may be greater than the energy available from the microgrid's production, storage systems and currently engaged contracts. In this case, the microgrid can buy additional energy e^t from the partner in order to fulfill unforeseen demand. We also assume a dissipation system is available with no cost of use associated.

We denote by MIP(DCCP) the formulation defined by objective function (2.1), constraints (2.4)–(2.12) and appropriated integrality and bounding constraints. With $|T|(|S_D| + 5|B| + |C|)$ constraints and $2|C| + |T||S_D| + 3|T||B| + |T|$ variables, this formulation is classified as a Mixed-Integer Programming (MIP) model [150], whose solution can be obtained with both commercial and open-source solvers, using well-known branch-and-bound algorithms. As seen in the experiments with case study instances, the solution to the deterministic CCP is returned by CPLEX solver in less than a second.

2.5 A robust formulation of the CCP

We consider in this section the robust version of the CCP, denoted as RCCP, in which the uncertainty of non-drivable systems will be treated via Robust Optimization methods. The developed model is capable of protecting against the worst-case realization of production and consumption of electricity, within a provided uncertainty set, considering all *uncertain non-drivable systems*, denoted as $\widehat{S}_{ND} \subseteq S$. Once again, the model solution consists of a list of contracts to engage in each time period, for the whole time horizon.

Regarding the uncertainty in non-drivable devices' production/consumption, the only information required by the model are the lower and upper bound parameters, namely $\{\underline{P_s^t}, \overline{P_s^t}\}$, $\forall t \in T, \forall s \in \widehat{S}_{ND}$, which can be determined via inference schemes based on historical data:

$\forall\, t \in T, \forall\, s \in \widehat{S}_{ND}$:

$\underline{P_s^t}$	lower bound on energy consumption($<$0)/production($>$0) of s in the whole period t
$\overline{P_s^t}$	upper bound on energy consumption($<$0)/production($>$0) of s in the whole period t

Similarly to drivable and non-drivable devices, uncertain devices $s \in \widehat{S}_{ND}$ also have associated operational costs v_s, per energy unit consumed/produced.

2.5.1 Definition of the uncertainty sets

In this work, we adopt a min-max criterion to assess the cost of feasible solutions to the problem. This means that we look for a solution that is feasible for each attribution of the uncertain parameters

and that minimizes the cost function in the worst case scenario. The uncertain data are assumed to be varying in a given uncertainty set.

The formulation of the Robust Optimization model is connected with the definition of this uncertainty set and this definition depends on the suppositions made on the problem being solved. In our problem, the set $\mathbf{U}(\mathbf{t},\mathbf{s})$ describes how the uncertainty is defined.
$\forall\, t \in T, \forall\, s \in \widehat{S}_{ND}$:

$\widehat{P}_s^t \in \mathbf{U}(\mathbf{t},\mathbf{s})$	energy consumption(<0)/production(>0) of s in the whole time period t

Consider a vector $v \in \Re^{n\times m}$. This text uses the vector notation $v^i = (v_j^i; j = 1,\ldots,m)$ and $v_j = (v_j^i; i = 1,\ldots,n)$. Hence, $\underline{P}^t = (\underline{P_s^t}; s \in \widehat{S}_{ND})$ and $\underline{P}_s = (\underline{P_s^t}; t \in T)$. If we presume that each uncertain parameter belongs to an interval, i.e, $\mathbf{U}(\mathbf{t},\mathbf{s}) = [\underline{P_s^t}, \overline{P_s^t}]$, the *box uncertainty set* [128], denoted here by $\mathcal{U}_{box}$, can be defined as:

$$\mathcal{U}_{box} = \times_{s\in\widehat{S}_{ND}} U_s$$

where $U_s = [\underline{P_s}, \overline{P_s}], s \in \widehat{S}_{ND}$.

Assuming that the uncertain parameter belongs to an interval is equivalent to say that it lies between a mean value and peak values, i.e.,

$$\mathbf{U}(\mathbf{t},\mathbf{s}) = \{\widehat{P}_s^t = \bar{v}_s^t + \Delta_s^t \widehat{v}_s^t \mid -\mathbf{1} \le \Delta_s^t \le \mathbf{1}\},$$

with $\widehat{v}\,_s^t = (\overline{P_s^t} - \underline{P_s^t})/2$ and $\bar{v}\,_s^t = \underline{P_s^t} + \widehat{v}\,_s^t$.

Now suppose that, given all uncertain devices $s \in \widehat{S}_{ND}$ and time periods $t \in T$, at most Γ uncertain parameters $\widehat{P}_s^t$ may reach peak values in the whole time horizon. We can then define the *budget uncertainty set*, studied in [16] and largely applied [1, 91, 26]:

$$\mathcal{U}_\Gamma = \{\widehat{P} \in \mathcal{U}_{box} : \textstyle\sum_{s\in\widehat{S}_{ND}} \sum_{t\in T} \mid \Delta_s^t \mid \le \Gamma\}.$$

The purpose of the budget of uncertainty is to control the level of conservatism of the robust solution, in terms of deviations in the uncertain model parameters. It allows an intuitive interpretation for the decision maker, providing a trade-off between the nominal performance of the deterministic model and the risk protection of the most conservative model. Additionally, the obtained robust counterpart remains efficiently solvable, provided that the original nominal problem can be effectively solved.

In the context of the RCCP, with the objective of simplifying the model and the analysis of the obtained results, we opted for a single budget parameter Γ. It controls the total number of deviated parameters regarding both energy consumption ($\widehat{P}_s^t < 0$) or production ($\widehat{P}_s^t > 0$) of all uncertain devices $s \in \widehat{S}_{ND}$, over all time periods $t \in T$.

2.5.2 Discussing the uncertainty definition

It is expectable to have the worst case scenario defined by the lowest possible production and/or by the highest possible consumption of systems in set $\widehat{S}_{ND}$. From Example 2.5.1, we can see that the opposite can also happen, i.e., the worst case scenario defined by the uncertain parameters can

also be associated with the highest possible production of an uncertain system $s \in \widehat{S}_{ND}$.

Example 2.5.1. *Let $T = \{0\}$, $C = C_0 = \{c_1\}$, $S_{ND} = \{s_1\}$, $S_D = B = \emptyset$ and $\widehat{S}_{ND} = \{s_2\}$. The contract offered by the partner has costs $v^t_{c_1}, \alpha^t_{c_1} > 0$, minimal and maximal consumption $\Pi^-_{c_1} = \Pi^+_{c_1} = \bar{a}$. The cost of buying energy not provided by contract c_1 is $\beta^0 > (v^t_{c_1} + \bar{a}\alpha^t_{c_1})/\bar{a}$. The certain and uncertain systems s_1 and s_2 are, respectively, a consumption and a production system. System s_1 has a cost $v_{s_1} = 0$ and a consumption $P^0_{s_1} = -\bar{a}$. System s_2 has a cost $v_{s_2} > (v^t_{c_1} + \bar{a}\alpha^t_{c_1})/\bar{a}$ and an uncertain production $\widehat{P}^0_{s_2} \in [0, \bar{a}]$. Consider $\beta^0 < v_{s_2}$. The optimal solution is $y_{c_1} = 0$. The worst case scenario is given by $\widehat{P}^0_{s_2} = \bar{a}$ with a cost equal to $\bar{a}v_{s_2}$. If $\beta^0 > v_{s_2}$, the optimal solution is also $y_{c_1} = 0$, but the worst case scenario is given by $\widehat{P}^0_{s_2} = 0$ with a cost of $\bar{a}\beta^0$.*

Situations like the one depicted in this small example (highest production/lowest consumption of non-drivable uncertain systems) can cause the client to engage contracts in order to sell the excess energy produced by the systems.

2.5.3 Robust counterpart

Similarly to the DCCP, we describe a formulation for the robust version of the problem based on time decomposition, in which decisions are made for every time period I_t. The Min-Max Adjustable Robust Counterpart (ARC) formulation, used in this work, ensures feasibility of the constraints for any realization of the uncertainty, through the appropriate selection of the second stage decision variables. For more details on Adjustable RO for multi-stage optimization problems, we refer the interested reader to [49].

In the robust version of our problem, the RCCP, variables y (defined in Section 2.4.2) are non-adjustable ones, i.e, they consist of “here and now” decisions, or first-stage variables, before having any knowledge of the actual value taken by the uncertainty. The other variables, namely q, r, h, g, x and e, are adjustable ones, i.e, they consist of “wait and see” decisions (i.e., second-stage variables) and define a set of decisions that depend on the uncertain parameters.

The Min-Max ARC, based on formulation MIP(DCCP) from Section 2.4.2, is as follows:

$$\text{(ARC)}\ \min_{y_c^t, E} \quad E \tag{2.13}$$

$$\text{s.t. } E \geq \sum_{t\in T}\sum_{c\in C_t}(v_c^t y_c^t + \alpha_c^t q_c^t(\widehat{P}_t)) + \sum_{s\in S_D} v_s \sum_{t\in T} P_s^t x_s^t(\widehat{P}_t) + \sum_{s\in \widehat{S}_{ND}} v_s \sum_{t\in T} \widehat{P}_s^t + \sum_{s\in B} v_s \sum_{t\in T}(g_s^t(\widehat{P}_t) + h_s^t(\widehat{P}_t)) + \sum_{t\in T}\beta^t e_t(\widehat{P}_t), \forall\, \widehat{P} \in \mathcal{U}, \tag{2.14}$$

$$\sum_{c\in C_t} q_c^t(\widehat{P}_t) + \sum_{s\in S_{ND}} P_s^t + \sum_{s\in \widehat{S}_{ND}} \widehat{P}_s^t + \sum_{s\in S_D} P_s^t x_s^t(\widehat{P}_t) + \sum_{s\in B}\lambda_s h_s^t(\widehat{P}_t) + e^t(\widehat{P}_t) \geq \sum_{s\in B} g_s^t(\widehat{P}_t), \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T, \tag{2.15}$$

$$h_s^t(\widehat{P}_t) \leq r_s^t(\widehat{P}_{t-1}), \forall \widehat{P} \in \mathcal{U}, \forall t \in T, \forall s \in B, \tag{2.16}$$

$$r_s^0 = u_s^0, \forall\, s \in B, \tag{2.17}$$

$$u_s^{min} \leq r_s^t(\widehat{P}_{t-1}) \leq u_s^{max}, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T \cup \{\bar{t}\},\ \forall\, s \in B, \tag{2.18}$$

$$r_s^{t+1}(\widehat{P}_t) = r_s^t(\widehat{P}_{t-1}) - h_s^t(\widehat{P}_t) + \lambda_s g_s^t(\widehat{P}_t), \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T,\ \forall\, s \in B, \tag{2.19}$$

$$g_s^t(\widehat{P}_t) \leq \theta_s^{abs}\delta^t, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T,\ \forall\, s \in B, \tag{2.20}$$

$$h_s^t(\widehat{P}_t) \leq \theta_s^{ref}\delta^t, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T,\ \forall\, s \in B, \tag{2.21}$$

$$\Pi_{t,c}^- y_c^t \leq q_c^t(\widehat{P}_t) \leq \Pi_{t,c}^+ y_c^t, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T,\ \forall\, c \in C_t : \Pi_{t,c}^+ > 0, \tag{2.22}$$

$$\Pi_{t,c}^+ y_c^t \leq q_c^t(\widehat{P}_t) \leq \Pi_{t,c}^- y_c^t, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T,\ \forall\, c \in C_t : \Pi_{t,c}^- < 0, \tag{2.23}$$

$$\sum_{t'\leq t} P_s^{t'} x_s^{t'}(\widehat{P}_t) \geq P^{min^t}{}_s, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T,\ \forall\, s \in S_D : P_s^t > 0, \tag{2.24}$$

$$\sum_{t'\leq t} P_s^{t'} x_s^{t'}(\widehat{P}_t) \leq P^{min^t}{}_s, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T,\ \forall\, s \in S_D : P_s^t < 0, \tag{2.25}$$

$$y_c^t \in \{0,1\}, \forall\, t \in T, \forall\, c \in C_t, \tag{2.26}$$

$$r_s^t(\widehat{P}_{t-1}), h_s^t(\widehat{P}_t), g_s^t(\widehat{P}_t) \geq 0, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T, \forall\, s \in B, \tag{2.27}$$

$$e^t(\widehat{P}_t) \geq 0, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T, \tag{2.28}$$

$$0 \leq x_s^t(\widehat{P}_t) \leq 1, \forall\, \widehat{P} \in \mathcal{U}, \forall\, t \in T, \forall\, s \in S_D, \tag{2.29}$$

where $\mathcal{U}$ is the uncertainty set chosen. For a given $t' \in T$, variables $q^{t'}$, $h^{t'}$, $g^{t'}$, $x^{t'}$, and $e^{t'}$ depend on the vector of uncertain parameters $\widehat{P}_{t'}$ while variables $r^{t'}$ depend on $\widehat{P}_{t'-1}$.

In a previous work [87], a conservative $\mathcal{U}_{box}$ uncertainty set was assumed and preliminary computational results were obtained. In the next subsection, we will explain how the dualization approach was employed to derive a robust counterpart for the $\mathcal{U}_\Gamma$ uncertainty set. Notice that the cost associated with the use of *uncertain non-drivable systems*, given by $\sum_{s\in\widehat{S}_{ND}} v_s \sum_{t\in T} \widehat{P}_s^t$, must be included in the robust model. Different from the cost of certain non-drivable systems, $\widehat{P}_s^t$ values are not constant and vary with the uncertain parameters. Also notice that constraints (2.19) can be used to eliminate variables h_s^t, reducing the set of second-stage variables.

An approach proposed in the literature to make model (ARC) tractable consists of restricting the functional relations q_c^t, r_s^t, g_s^t, x_s^t and e^t to be affine by replacing them with linear decision rules (LDR) [13]. Also known as affine policies, LDRs have been commonly used in the literature as an effective approximation to multistage RO problems [63, 146, 91], with each recourse decision taking the form of an affine function of the uncertain parameters.

This way, we restrict recourse variables, say $g_s^t(\widehat{P}^t)$, to be affinely dependent on the primitive uncertainties, considering all uncertain devices $\varsigma \in \widehat{S}_{ND}$ and all time periods prior to t. Of course,

only in very rare occasions, linear decision rules are optimal. Indeed, the main motivation for linear decision rules is its tractability. The following decision rules were applied for the set of adjustable variables in our problem:

$$r_s^t = r_{ts}^0 + \sum_{\tau=0}^{t-1} \sum_{\varsigma \in \widehat{S}_{ND}} r_{ts}^{\tau\varsigma} \widehat{P}_\varsigma^\tau, \forall t \in T \setminus \{0\} \cup \{\bar{t}\}, \forall\, s \in B, \tag{2.30}$$

$$g_s^t = g_{ts}^0 + \sum_{\tau=0}^{t} \sum_{\varsigma \in \widehat{S}_{ND}} g_{ts}^{\tau\varsigma} \widehat{P}_\varsigma^\tau, \forall t \in T, \forall\, s \in B, \tag{2.31}$$

$$x_s^t = x_{ts}^0 + \sum_{\tau=0}^{t} \sum_{\varsigma \in \widehat{S}_{ND}} x_{ts}^{\tau\varsigma} \widehat{P}_\varsigma^\tau, \forall t \in T, \forall\, s \in S_D, \tag{2.32}$$

$$q_c^t = q_{tc}^0 + \sum_{\tau=0}^{t} \sum_{\varsigma \in \widehat{S}_{ND}} q_{tc}^{\tau\varsigma} \widehat{P}_\varsigma^\tau, \forall t \in T, \forall\, c \in C_t, \tag{2.33}$$

$$e^t = e_t^0 + \sum_{\tau=0}^{t} \sum_{\varsigma \in \widehat{S}_{ND}} e_t^{\tau\varsigma} \widehat{P}_\varsigma^\tau, \forall t \in T. \tag{2.34}$$

As seen in the next subsection, after bringing the linear decision rules to the formulation, by taking $\mathcal{U} = \mathcal{U}_\Gamma$, each constraint holding uncertain parameters is transformed by means of strong duality theory. As a result, we obtain a linear approximation to the model, called (ARC-L1). Each inequality will be characterized in terms of max/min values, and later replaced by its corresponding dual equivalent. In this process, a new set of continuous variables and a new set of constraints will be added to the formulation. This final product is a MILP model to the robust problem, which will be called MIP(RCCP), and whose solution can be obtained with commercial optimization solvers.

2.5.4 RCCP under budgeted uncertainty

Given the definition of the budget uncertainty set $\mathcal{U}_\Gamma$, we now present a dualization rule that can be applied to each constraint of the robust counterpart. Each ARC constraint can be written either in form (a) $f^0 + \sum_{t\in T} \sum_{s\in \widehat{S}_{ND}} \widehat{P}_s^t f_s^t \leq 0$, or (b) $f^0 + \sum_{t\in T} \sum_{s\in \widehat{S}_{ND}} \widehat{P}_s^t f_s^t \geq 0$, where f^0 is the independent term and f_s^t is the term that depends on the uncertain parameters $\widehat{P}_s^t \in \mathcal{U}_\Gamma$. In form (a), each problem constraint can be rewritten as:

$$f^0 + \max_{\widehat{P}\in\mathcal{U}_\Gamma} \left\{ \sum_{t\in T} \sum_{s\in \widehat{S}_{ND}} \widehat{P}_s^t f_s^t \right\} \leq 0, \tag{2.35}$$

with the second term being reformulated as:

$$\max_{\Delta_{t,s}^+,\, \Delta_{t,s}^-} \Big\{ \sum_s \sum_t f_s^t \left(\overline{v}_s^t + \Delta_{t,s}^+ . \widehat{v}_s^t - \Delta_{t,s}^- . \widehat{v}_s^t \right) : \tag{2.36}$$

$$\Delta_{t,s}^+ + \Delta_{t,s}^- \leqslant 1,\ s \in \widehat{S}_{ND}, t \in T, \quad (\mu_s^t) \tag{2.37}$$

$$\sum_s \sum_t \left(\Delta_{t,s}^+ + \Delta_{t,s}^- \right) \leqslant \Gamma, \qquad (\rho) \tag{2.38}$$

$$\Delta_{t,s}^+ \geqslant 0,\ \Delta_{t,s}^- \geqslant 0,\ s \in \widehat{S}_{ND}, t \in T \Big\} \leqslant 0 \tag{2.39}$$

Where variables $\Delta_{t,s}^+$ and $\Delta_{t,s}^-$ indicate that the uncertain parameter $\widehat{P}_s^t$ has oscillated above (or below) its nominal value $\overline{P}_s^t$. Constraints (2.37) limit the oscillation according to the maximum value allowed ($\widehat{v}_s^t$) and constraints (2.38) limit the budget of uncertainty to Γ.

Analogously, for each problem constraint in form (b), we derive:

$$\min_{\Delta_{t,s}^+,\, \Delta_{t,s}^-} \left\{ \sum_s \sum_t f_s^t \left(\overline{v}_s^t + \Delta_{t,s}^+ . \widehat{v}_s^t - \Delta_{t,s}^- . \widehat{v}_s^t \right) : \text{s.t. (2.37)-(2.39)} \right\} \geqslant 0. \tag{2.40}$$

The inclusion of the above robustified equations in the tractable MILP model is possible via

dualization. In both cases above, dual variables μ_s^t and ρ can be derived, along with dual objective function: $\sum_s \sum_t \mu_s^t + \rho.\Gamma + \sum_s \sum_t f_s^t.\overline{v}_s^t$.

In the first case (a), the dual constraints are:

$$\mu_s^t + \rho \geqslant |\widehat{v}_s^t.f_s^t| \equiv \begin{cases} \mu_s^t + \rho \geqslant \widehat{v}_s^t.f_s^t \quad (\Delta_{t,s}^+) & (2.41) \\ \mu_s^t + \rho \geqslant -\widehat{v}_s^t.f_s^t \ (\Delta_{t,s}^-) & (2.42) \\ \mu_s^t \geqslant 0, \ \rho \geqslant 0 & (2.43) \end{cases}$$

And, in the second case (b), the dual constraints are:

$$\mu_s^t + \rho \leqslant |\widehat{v}_s^t.f_s^t| \equiv \begin{cases} \mu_s^t + \rho \leqslant \widehat{v}_s^t.f_s^t \quad (\Delta_{t,s}^+) & (2.44) \\ \mu_s^t + \rho \leqslant -\widehat{v}_s^t.f_s^t \ (\Delta_{t,s}^-) & (2.45) \\ \mu_s^t \leqslant 0, \ \rho \leqslant 0 & (2.46) \end{cases}$$

In form (a), after dualization, each constraint is replaced by the following constraints in the tractable MILP model:

$$f^0 + \sum_s \sum_t \mu_s^t + \rho.\Gamma + \sum_s \sum_t f_s^t.\overline{v}_s^t \leqslant 0 \ , \text{ and (2.41)-(2.43).}$$

While, in form (b), each constraint is replaced by constraints:

$$f^0 + \sum_s \sum_t \mu_s^t + \rho.\Gamma + \sum_s \sum_t f_s^t.\overline{v}_s^t \geqslant 0 \ , \text{ and (2.44)-(2.46).}$$

The resulting tractable robust MILP model has $(5 + |T||\widehat{S}_{ND}|)(2|C| + |T||S_D| + 3|T||B| + |T|)$ constraints and $|T|^2|\widehat{S}_{ND}|(2|B| + |C| + 1) + (6|T||S_D| + 8|C| + 14|T||B| + |T||C|)$ variables.

2.6 Real-time energy scheduling with the RTCS

The solution of the CCP (either the deterministic version in Section 2.4 or the robust version from Section 2.5) provides the client a decision for the first level in our framework: the contract subscription for the whole time horizon. However, for the success of the proposed contract framework, an efficient real-time energy scheduling mechanism is needed, so that distributed energy resources, storage devices and drivable loads within the microgrid are operated in a coordinated and coherent way, together with the energy exchange with the main grid.

Recall the RTCS definition given in Section 2.2. In order to perform energy scheduling, each time period I_t is further subdivided into δ^t **time slots of size** Δ. At this time scale, the instantaneous production/consumption of each microgrid system is measured. For *certain* (*drivable* and *non-drivable*) systems, the consumption/production, for each time period I_t, is known beforehand. And every Δ time units a scheduling decision has to be made by the control device, according to the state of the microgrid.

In this sense, the RTCS consists of a scheduling heuristic that, based on the microgrid state and the energy contracts engaged by the client, solves an online optimization problem, selecting in real-time the set of actions to be taken, with the objective of reducing energy consumption costs and promoting load balance, while, at the same time, dealing with the effects of uncertainty. Remark that

the solution approaches commonly used to solve off-line scheduling problems are not appropriate for the on-line scheduling case.

Since the RTCS operates according the Contract Collaboration Framework, we remark that, once the heuristic starts, the set of subscribed energy contracts $\mathbf{y}_0$, for the whole time horizon, has already been established. Such decision is made by the client after running one of the previously presented CCP models. According to the microgrid energy balance, the following operations must be considered: (a) turn on/off a production/consumption *drivable system*; (b) buy/sell a quantity of energy under an engaged contract; (c) recharge/retrieve energy from a *storage system*; (d) buy energy out of engaged contracts; (e) throw energy away (if remaining energy cannot be sold back to the grid). Also, a subset of actions taken in a specific moment must obey the Contract Collaboration Problem constraints.

2.6.1 Naïve RTCS policy

This section describes the most straightforward approach to perform energy scheduling. The proposed naïve control strategy can be used as a baseline strategy to schedule the use of the Battery Energy Storage Systems (BESS) and the energy exchange via contracts. The naïve strategy relies on two assumptions. First, the microgrid should only sell energy via contracts as a last resort because the selling price is typically lower than the contract buying price. Besides, most of the time, the production obtained from renewables does not match the microgrid's energy demand. As a result, the microgrid will eventually buy energy from the power grid.

The proposed naïve RTCS policy works as follows. At each time step, the current demand for electricity is determined as the sum of the amount of energy required by the consumer drivable and non-drivable systems minus the amount of energy actually produced by the microgrid. In case the produced energy outweighs the demand, the resulting surplus is used to charge the battery. If the BESS is already full or the surplus exceeds the charging rate, the excess energy is sold via contracts. On the contrary, if existing demand goes beyond the available produced energy, the difference is provided by discharging the battery. If the stored energy is not enough, then the remaining required energy is bought from the grid, first via available engaged contracts (provided they still have existing capacity), or bought out of any engaged contracts, possibly at a higher price.

2.6.2 Using model solution as a look-ahead policy to guide RTCS

The solution of the CCP models provide not only a list of contracts to engage, but also a set of values that can be used as a look-ahead (LA) policy to guide the RTCS energy dispatch operations, at each time period I_t. The policy is defined by the optimal value of all model variables, except y variables.

When using the deterministic model, look-ahead values are obtained directly from the model

solution, while for the robust models, the LDRs (2.30)-(2.34) are used to derive the look-ahead values for the current time period $I_{t'}$. In Section 2.7, we will show that the look-ahead policies based on the robust models can effectively enhance the performance of the RTCS. Among the advantages, the better utilization of storage devices and greater protection against uncertainty, when compared to the deterministic model.

We propose different heuristics for the RTCS, each one with a distinct behavior. As previously mentioned, the RTCS operates based on a predefined set of engaged contracts, obtained from a specific CCP model solution. Therefore, in the remainder of this section, we refer to X-RTCS as the general RTCS that can be executed based on an existing CCP model solution X.

Algorithm 1 depicts the X-RTCS executed inside a given time period $I_{t'}$, every Δ time units or, analogously, at each time slot $d \in \{1, \ldots, \delta^{t'}\}$. At this time, the uncertainty concerning energy production/consumption has been revealed for all time periods before $I_{t'}$, and the microgrid configuration is given by its battery storage levels, drivable system requirements, load demand and renewable production. Let $X(\mathbf{y}_0, t')$ be the X model obtained by fixing $y = \mathbf{y}_0$ and all variables and parameters related with $t < t'$. The optimal solution of $X(\mathbf{y}_0, t')$ serves as a policy for all $t \geq t'$. Two different parameters, named REOPTIMIZE and GAP POLICY, define how the $X(\mathbf{y}_0, t')$ solution will be used as a policy in the X-RTCS heuristic.

The first one concerns the usage of model X. The REOPTIMIZE option determines which model X solution will be used as a forecast tool to determine the initial quantities regarding how much energy will be consumed or how much excess energy will be sent to the grid, via engaged contracts [1], and how long *drivable systems* should remain powered on. If REOPTIMIZE is enabled (line 1), model X will be reoptimized at each time period $I_{t'}$, for the remaining time periods $t \geq t'$ (line 2), and the new solution obtained will guide the initial energy quantities (line 3). Otherwise, as listed on line 5, the RTCS policy will be based on the initial solution at the start of the time horizon ($t = 1$), i.e., the optimal solution of $X(\mathbf{y}_0, 1)$.

The next step of the algorithm is the calculation of the current microgrid energy gap (i.e., total consumption *minus* total production) at the current time slot d (line 9), based on collected data regarding instantaneous energy production/consumption, minimum contract engagements, as well as model predictions regarding batteries and drivable systems (lines 6-8). We denote by $q_c^{t',d}$ the amount of energy that will be bought or sold via the contract c in time slot d; and $x_s^{t',d}$ as the percentage of time in which drivable system s will be on at time slot d. Also assume $\widehat{P}_s^{t',d}$ is an estimation of the uncertain production/consumption of device s at time slot d.

Finally, according to this information, X-RTCS must decide which additional dispatch operations will be executed to balance supply and demand, by applying a GAP POLICY (Table 2.2). These operations involve, for example, seling (line 15) or buying (line 24) additional energy to/from the

1. It is worth noting, however, that the amount of electricity bought or sold via each engaged contract may be more than the initial values proposed by the model policy, depending on the actual energy demand.

partner, turning on/off drivable systems (lines 16&23) and interacting with an energy storage system (lines 14&25).

Gap	**Cheapest gap policy**
Positive	Cheapest storage operation (batteries/drivable/contracts)
Negative	Cheapest retrieval operation (batteries/drivable/contracts)
Gap	**Conservative gap policy**
Positive	Store energy surplus in batteries, then use sell contracts
Negative	Buy from engaged contracts first, then use batteries

Table 2.2 – X-RTCS policy executed at each time slot d (inside period $I_{t'}$), according to the microgrid energy gap, defined as $\sum(\text{Production}) - \sum(\text{Consumption})$. A positve gap $(+)$ means there is energy surplus in the current time slot, while a negative gap $(-)$ represents lack of energy (more power needs to be bought from the partner or produced by the microgrid).

In summary, the combination of REOPTIMIZE and GAP POLICY parameters yield 4 different look-ahead heuristics, whose behavior is determined by which CCP model predictions are used (from reoptimized model or not), along with a strategy to either fulfill demands greater than the microgrid's own production (negative gap) or use the available energy surplus (positive gap).

Remark that the algorithm considers the cost of purchasing additional blocks of energy (negative imbalance) and the revenue from selling surplus energy (positive imbalance) due to deviations concerning the forecast of energy contract usage, made in the beginning of X-RTCS. Additionally, the amount of electricity bought out of any contract $e(\cdot)$ presumes an indirect penalization on price, since they are less attractive than settled day-ahead prices.

2.7 Experimental results

This section sets up a realistic microgrid and conducts simulations with different sets of scenarios, comprising uncertain electricity production and consumption in a given time horizon. The main objective is to evaluate the impact of adopting a robust approach for engaging in flexible energy contracts. This is achieved through the performance assessment of the RTCS approaches proposed in the previous section, based on contract decisions taken by either the deterministic or the robust CCP model solution.

2.7.1 Computational environment and simulation details

The mathematical models and numerical simulations were coded in Julia 1.6.0 using CPLEX solver 20.1.0, and their source code is available at https://github.com/levorato/ccp_rtcs. All experiments were performed on a workstation with an Intel Xeon CPU X5355 $\times$ 8 with 64 GB RAM, under Ubuntu 18.04 LTS. As defined in Section 2.6.2, each RTCS heuristic can use either

Algorithm 1: RTCS algorithm that runs at each time slot $d \in \{1, \ldots, \delta^{t'}\}$, inside time period $I_{t'}$.

Input: CCP model CCPM, initial CCPM solution $X(\mathbf{y}_0, 1)$, GAP POLICY, REOPTIMIZE
Result: Set of policies $q_c(\cdot), x_s(\cdot), h_s(\cdot), e(\cdot)$

if REOPTIMIZE *is enabled* **then**
  Reoptimize CCPM with $t_0 := t'$ and fix engaged contracts $\overline{\mathbf{y}} = \mathbf{y}_0$
  Let $\{q^{t'}, x^{t'}, g^{t'}, h^{t'}\}$ be the reoptimized CCPM solution at $t = t_0 = t'$
else `// Use initial CCP model solution at` $t_0 = 1$
  Let $\{q^{t'}, x^{t'}, g^{t'}, h^{t'}\}$ be the initial CCPM solution at $t = t'$
$q_c^{t',d} := \lceil q_c^{t'} / \delta^{t'} \rceil, \forall c \in C_{t'}$ `// Initial contract usage according to` $\mathbf{y}_0$
$x_s^{t',d} := \max\left[x_s^{t'}, \frac{P^{min}{}_s^{t'}}{P^{max}{}_s^{t'}}\right], \forall s \in S_D$ `// Power drivable according to CCP solution`
$h_s^{t',d} := \lceil h_s^{t'} / \delta^{t'} \rceil;\ g_s^{t',d} := \lceil g_s^{t'} / \delta^{t'} \rceil$ `// Use batteries according to CCP solution`
`/* Sum energy consumption/production for all devices (certain and uncertain) */`
$gap := \sum_{s \in \widehat{S}_{ND}} \widehat{P}_s^{t',d} + \sum_{s \in S_{ND}} \frac{P_s^{t'}}{\delta^{t'}} + \sum_{c \in C_{t'}} q_c^{t',d} + \sum_{s \in S_D} \frac{x_s^{t',d} P_s^{t'}}{\delta^{t'}} + \sum_{s \in S_B} (h_s^{t',d} - g_s^{t',d})$
if $gap > 0$ **then** `// energy left over`
  if GAP POLICY = *cheapest* **then**
    Execute dispatch operations (Charge batteries, Sell via contracts, Power drivable consumers) following smallest energy cost first
  else `// conservative GAP POLICY`
    gap := `Charge-batteries`$(\{s \in S_B\})$
    gap := `Sell-energy-surplus-via-contracts`$(\{c \in C_{t'} : \Pi^-_{t',c} < 0\})$
    gap := `Power drivable consumers`$(\{s \in S_D : P^{min}{}_s^{t'} < 0\})$
  $e_{t',d} := 0$
  Throw remaining energy away
else `// Negative gap, need for additional energy`
  if GAP POLICY = *cheapest* **then**
    Execute operations (Turn on drivable producer, Use batteries, Consume from contracts, Consume out of contract) according to smallest cost first
  else `// conservative GAP POLICY`
    gap := `Turn-on-drivable-producer-devices`$(\{s \in S_D : P^{min}{}_s^{t'} > 0\})$
    gap := `Consume-energy-via-contracts`$(\{c \in C_{t'} : \Pi^-_{t',c} > 0\})$
    gap := `Discharge-batteries`$(\{s \in S_B\})$
  $e_{t',d} := gap$ `// Consume remaining energy out of contracts if needed`

the deterministic or the robust CCP model solution as input. We denote by **Det-RTCS** the RTCS based on the deterministic model, and **Rob-RTCS** the one based on the robust budgeted model. The RTCS simulation is based on sets of realistic scenarios from the case study defined in this section.

Remember that, in order to obtain a deterministic solution for the CCP model, it is necessary to establish a fixed value for the uncertain parameters. In this study, we solved the deterministic model by using three sets of values for uncertain parameters $\widehat{P}_s$ of each system $s \in \widehat{S}_{ND}$.

2.7.2 Microgrid in a research building in Tsukuba, Japan

This case study involves the microgrid of a research building in Tsukuba, Japan. A multiyear dataset [140] provides microgrid statistics in full details (every second) and summarized (per hour), for the period between april 2015 and april 2018. Supplied data includes the Battery Energy Storage System (BESS) installed (active power, voltage, current, state of charge), the power generation from the four operating solar arrays, as well as purchased electricity (voltage, active power), solar irradiance, list of holidays and electricity prices (including surcharges).

Four problem instances were generated, one for each season of the year. Thus the lower and upper bounds for uncertain consumption and production were calculated as a function of the historical data for the corresponding season.

2.7.3 Problem instance generation

The considered planning horizon comprises 24 time periods of 1 hour, each one with $\delta^t = 6$. Given all periods, a total of 457 contracts were proposed, inspired by Électricité de France price distribution [33], allowing the client to buy electricity from the partner at different quantities and costs.

Concerning uncertain devices demand and production of electricity at each time period t, historical data of the microgrid is used to calculate the lower and upper bounds of these values. Moreover, instead of applying simple min/max approach, we use the 10th and 90th quantiles to determine the P_{min} and P_{max} values, which guarantees robustness against outliers. The determination of BESS kWh price is based on the cost model of [18].

2.7.4 RTCS simulation and scenario types

Based on the production and consumption history of the Tsukuba microgrid, the simulation objective is twofold: to evaluate how the solutions provided by the two CCP models proposed (robust and deterministic) behave under uncertainty, and to assess the performance of the RTCS policies defined in Section 2.6.2.

For each *uncertain system*, a particular scenario contains the realization of the uncertain electricity production (or consumption) values for each time slot over the whole time horizon. Given a list of engaged contracts (previously obtained with the solution of the CCP model), the simulation iterates over each time step, executing the chosen RTCS policy. As explained in the previous section, at this point, real-time energy scheduling actions are taken, according to the current state of the microgrid and the realization of energy production and consumption values. Among these values, the ones concerning the *uncertain systems* are obtained through the given scenario data in the current simulation.

For the Tsukuba microgrid, an individual set of scenarios was generated for each of the four seasonal instances, based on real values of PV production and building consumption provided in the dataset. The spring instance scenarios, for example, encompass all information recorded between March 20th until June 21st, given the dataset's 3-year time horizon.

In summary, for each microgrid instance, the simulation process consists in testing the RTCS policies based on each CCP model solution. The combination of 2 types of gap policy (Cheapest and Conservative), with or without model reoptimization, yields a total of 4 possible X-RTCS procedures for each model X, deterministic or robust. Simulation is then performed by executing each pair of model X and X-RTCS heuristic on the proposed microgrid instances and their associated scenario groups. For each of the 4 seasonal Tsukuba instances, simulation will be executed on a specific scenario set from each season.

2.7.5 Performance of the robust solution method

	Deterministic / Φ			Robust budget / Γ					
Instance	0	50	100	0	20	40	60	80	100
Autumn	19,969.98 0.07 s	26,273.60 0.25 s	32,531.61 0.36 s	25,984.11 0.71 s	29,981.41 189.20 s	31,527.87 209.69 s	32,186.66 86.03 s	32,300.27 64.91 s	32,302.23 57.86 s
Spring	18,345.28 0.15 s	24,429.74 0.31 s	30,534.44 0.32 s	24,161.60 0.30 s	27,899.25 241.17 s	29,398.48 55.88 s	30,125.96 81.47 s	30,294.75 40.48 s	30,294.95 50.57 s
Summer	22,281.03 0.33 s	29,954.69 0.29 s	37,364.42 0.32 s	29,611.22 0.84 s	34,129.95 2159.94 s	36,082.87 738.91 s	36,960.79 603.36 s	37,132.68 294.62 s	37,132.68 263.40 s
Winter	20,443.66 0.04 s	27,183.29 0.25 s	33,728.24 0.28 s	26,895.68 0.68 s	31,111.05 229.54 s	32,670.23 68.91 s	33,394.60 75.76 s	33,482.34 60.53 s	33,482.90 54.43 s

Table 2.3 – Robust *vs.* Determinisitc model result comparison for different budget parameters. The first value indicates the objective function value obtained, followed by the time spent (in seconds) to obtain the optimal solution. CPLEX default optimality gap of 10^{-4} was applied.

The study performed in this section analyses several cost and reliability metrics obtained from simulations of the proposed RTCS policies. By mimicking the real-time operation of the microgrid energy management system, each simulation was based on a specific solution provided by either the robust or the deterministic CCP model.

Instance / RTCS Policy		Deterministic / Φ			Robust budget / Γ					
	Variable	0%	50%	100%	0%	20%	40%	60%	80%	100%
autumn cheapest	Cost Avg ($)	94,758.0	27,638.0	31,791.0	49,191.6	34,128.8	27,919.7	29,490.5	40,486.3	31,938.6
	Cost Std ($)	55,760.3	3,077.8	477.1	26,486.9	6,898.7	2,182.1	1,772.7	12,741.9	4,977.2
	Cost CVaR ($)	179,591.2	32,662.8	32,382.9	92,132.1	45,220.2	31,554.8	32,396.3	60,409.8	39,906.4
	OC Cost Avg ($)	73,516.4	406.5	0.0	17,677.2	5,623.2	99.0	102.2	12,506.7	3,540.6
	Penalty Freq (%)	76.4	3.5	0.0	16.3	5.8	0.3	0.3	14.2	3.4
	SOC Avg (%)	38.2	67.7	91.7	68.2	69.8	68.2	73.4	71.9	72.5
	SOC Std (%)	12.2	28.8	19.0	28.1	29.4	29.6	29.3	29.7	29.7
autumn cheapest+ReOpt	Cost Avg ($)	94,486.2	27,367.2	31,223.2	46,781.5	31,941.4	27,338.6	27,530.0	39,762.5	30,758.4
	Cost Std ($)	55,872.6	3,271.5	622.1	26,155.9	7,286.9	2,530.4	2,453.3	13,421.8	5,539.0
	Cost CVaR ($)	179,581.4	32,607.8	32,091.1	90,519.4	43,902.0	31,479.3	31,510.4	60,727.0	39,705.3
	OC Cost Avg ($)	73,128.5	406.5	0.0	15,949.0	5,121.0	99.0	99.0	13,163.7	3,457.0
	Penalty Freq (%)	75.8	3.5	0.0	15.9	6.4	0.3	0.3	16.2	4.0
	SOC Avg (%)	40.2	68.4	91.6	69.1	68.7	69.9	70.9	70.5	74.2
	SOC Std (%)	14.3	27.3	18.2	26.5	27.6	27.7	28.5	26.9	26.9
autumn conservative	Cost Avg ($)	93,504.6	27,858.4	31,849.2	47,212.1	33,243.9	28,002.5	29,553.9	39,614.5	31,312.5
	Cost Std ($)	56,354.0	3,004.8	555.6	25,330.0	6,529.3	2,117.0	1,738.0	11,875.6	4,533.3
	Cost CVaR ($)	179,582.4	32,682.6	32,650.3	90,514.8	44,291.2	31,461.5	32,398.7	59,054.6	38,917.0
	OC Cost Avg ($)	72,042.4	392.2	0.0	15,911.4	4,617.2	78.6	79.9	11,514.0	2,795.1
	Penalty Freq (%)	77.8	3.6	0.0	18.4	7.6	0.3	0.3	16.3	4.0
	SOC Avg (%)	43.4	77.7	93.4	71.9	75.7	78.7	81.9	75.7	80.1
	SOC Std (%)	19.8	27.1	16.1	27.4	27.5	25.5	24.3	27.0	25.4
autumn conservative+ReOpt	Cost Avg ($)	93,207.7	27,552.1	31,300.5	45,754.8	31,276.0	27,424.1	27,561.9	37,929.5	30,126.7
	Cost Std ($)	56,490.5	3,187.0	709.6	25,098.8	6,823.9	2,460.5	2,479.6	12,288.9	5,000.4
	Cost CVaR ($)	179,577.0	32,662.5	32,374.0	88,850.8	42,866.4	31,477.5	31,600.6	58,263.8	38,498.1
	OC Cost Avg ($)	71,714.2	381.8	0.0	15,134.2	4,310.1	77.1	79.7	11,125.2	2,713.5
	Penalty Freq (%)	77.8	3.6	0.0	18.4	7.6	0.3	0.3	16.3	4.0
	SOC Avg (%)	44.7	80.2	93.8	73.2	75.3	81.8	81.6	76.0	82.3
	SOC Std (%)	21.1	25.5	15.7	25.9	26.4	24.2	24.6	25.3	23.8
autumn naïve	Cost Avg ($)	94,924.7	31,022.1	33,244.7	50,149.3	35,176.8	32,654.4	32,699.1	41,556.7	35,122.0
	Cost Std ($)	55,737.6	1,885.5	443.0	24,725.5	5,770.0	864.4	830.7	10,828.7	3,570.1
	Cost CVaR ($)	179,606.9	33,874.3	33,871.2	90,187.1	45,006.2	33,929.1	33,900.6	59,656.9	41,025.8
	OC Cost Avg ($)	73,858.1	280.1	0.0	17,219.9	4,297.9	42.2	38.2	10,412.0	2,650.5
	Penalty Freq (%)	77.8	3.2	0.0	14.4	5.5	0.1	0.1	14.1	3.3
	SOC Avg (%)	31.7	76.6	94.3	71.6	87.0	94.6	94.2	92.0	93.9
	SOC Std (%)	0.1	30.1	16.4	28.7	25.8	18.1	17.4	19.8	18.6
spring cheapest	Cost Avg ($)	82,186.0	49,631.0	29,808.7	104,241.0	30,524.7	26,182.7	27,256.6	28,428.7	28,471.7
	Cost Std ($)	44,527.0	12,653.1	310.9	32,307.5	4,890.4	1,823.3	2,598.0	831.8	1,643.7
	Cost CVaR ($)	144,445.2	68,773.0	30,209.9	147,380.1	38,129.8	29,027.2	31,491.0	29,776.1	31,088.5
	OC Cost Avg ($)	60,979.9	23,133.2	0.2	62,076.8	5,773.3	0.0	1,497.9	0.0	346.1
	Penalty Freq (%)	76.1	23.7	0.1	43.9	5.6	0.0	1.9	0.0	1.9
	SOC Avg (%)	36.1	58.6	91.6	36.3	58.7	61.1	58.1	72.6	67.0
	SOC Std (%)	10.5	27.6	17.0	8.9	28.0	28.3	28.8	29.0	29.1
spring cheapest+ReOpt	Cost Avg ($)	82,052.9	49,300.1	29,471.5	97,740.2	28,074.3	26,046.1	26,759.9	26,380.9	27,470.1
	Cost Std ($)	44,570.9	12,899.3	438.3	32,312.2	4,783.8	1,961.2	2,634.4	1,723.1	3,021.3
	Cost CVaR ($)	144,425.1	68,774.0	30,128.9	144,888.6	36,043.9	29,058.2	31,084.1	29,043.4	32,189.3
	OC Cost Avg ($)	60,764.1	23,014.2	0.2	55,391.8	4,274.0	0.0	1,066.2	0.0	1,191.8
	Penalty Freq (%)	75.5	23.9	0.1	43.9	5.5	0.0	2.3	0.0	2.7
	SOC Avg (%)	38.2	58.1	90.5	63.4	62.2	64.2	65.4	62.6	63.1
	SOC Std (%)	12.9	26.8	17.8	27.3	26.4	26.8	26.8	27.7	27.5
spring conservative	Cost Avg ($)	81,254.4	48,523.9	29,985.3	103,804.9	30,307.6	26,365.0	27,148.5	28,550.6	28,585.7
	Cost Std ($)	45,007.2	12,686.5	438.6	32,620.3	4,665.6	1,762.3	2,319.9	890.3	1,388.0
	Cost CVaR ($)	144,404.3	68,682.4	30,671.3	147,554.6	37,881.0	29,071.7	30,992.7	29,975.4	30,815.2
	OC Cost Avg ($)	59,869.0	22,171.6	0.2	61,622.5	5,214.9	0.0	1,011.7	0.0	356.2
	Penalty Freq (%)	76.9	25.7	0.1	43.9	6.9	0.0	2.3	0.0	3.6
	SOC Avg (%)	40.6	64.0	95.4	38.3	66.0	74.7	66.4	81.6	74.4
	SOC Std (%)	17.3	29.0	13.3	12.7	28.8	25.9	28.8	24.3	26.5
spring conservative+ReOpt	Cost Avg ($)	80,876.6	48,324.4	29,689.7	97,190.6	28,015.7	26,261.2	26,799.5	26,654.3	26,944.2
	Cost Std ($)	45,122.8	12,837.6	574.8	31,950.5	4,427.5	1,941.8	2,278.1	1,693.2	2,370.5
	Cost CVaR ($)	144,691.0	68,648.3	30,595.9	144,014.4	35,301.6	29,166.0	30,302.0	29,204.7	30,752.9
	OC Cost Avg ($)	59,446.2	22,154.8	0.2	54,914.1	3,936.4	0.0	626.5	0.0	490.8
	Penalty Freq (%)	76.9	25.7	0.1	43.9	6.9	0.0	2.3	0.0	3.6
	SOC Avg (%)	42.7	63.4	95.5	63.8	71.1	81.8	80.5	78.5	75.4
	SOC Std (%)	19.6	28.5	13.1	27.3	26.4	23.0	24.0	25.8	26.6
spring naïve	Cost Avg ($)	82,254.6	47,094.5	32,054.0	102,186.3	33,056.5	32,106.1	31,836.7	32,405.1	31,565.9
	Cost Std ($)	44,485.8	9,188.1	230.2	30,518.9	3,437.6	215.9	778.6	200.8	949.0
	Cost CVaR ($)	144,446.5	61,597.8	32,409.6	144,268.3	38,840.6	32,439.3	33,026.5	32,705.1	32,975.9
	OC Cost Avg ($)	61,162.3	19,660.5	0.0	58,143.1	4,547.8	0.0	422.8	0.0	292.0
	Penalty Freq (%)	76.8	23.3	0.0	41.4	4.7	0.0	1.3	0.0	1.9
	SOC Avg (%)	31.7	61.1	92.0	49.6	84.7	93.3	94.3	96.1	86.4
	SOC Std (%)	0.0	27.8	19.9	14.9	27.0	17.6	15.8	13.7	22.3

Table 2.4 – Robust *vs.* Deterministic RTCS performance comparison for autumn and spring scenario groups. *Cost Avg* is the average *scenario cost* over all simulations. *Cost Std* represents the standard deviation of *scenario cost*. *Cost CVaR* is the conditional value at risk of scenario cost at 80% confidence level (i.e., the average scenario cost of the 20% highest scenario costs). *OC Cost Avg* is the average cost from Out of Contract (OC) energy consumption. *Penalty Freq* is the proportion of time periods with OC consumption. *SOC Avg* and *SOC Std* are the average and standard deviation of BESS State Of Charge (SOC).

Instance / RTCS Policy		Variable	Deterministic / Φ 0%	Deterministic / Φ 50%	Deterministic / Φ 100%	Robust budget / Γ 0%	Robust budget / Γ 20%	Robust budget / Γ 40%	Robust budget / Γ 60%	Robust budget / Γ 80%	Robust budget / Γ 100%
summer	cheapest	Cost Avg ($)	125,414.5	31,732.4	36,431.7	53,796.2	52,252.5	33,014.1	60,577.1	33,982.6	33,113.3
		Cost Std ($)	54,258.0	3,317.8	369.2	17,148.2	16,370.9	1,874.9	21,842.2	1,440.8	1,814.3
		Cost CVaR ($)	204,123.3	37,232.9	36,896.1	79,873.1	75,714.4	36,167.0	92,785.6	36,483.6	36,171.1
		OC Cost Avg ($)	93,388.0	221.9	1.0	17,614.9	18,616.6	1.0	33,416.7	1.0	1.0
		Penalty Freq (%)	70.9	0.8	0.0	8.7	11.0	0.0	23.7	0.0	0.0
		SOC Avg (%)	34.3	63.4	92.4	58.1	63.3	68.4	62.9	74.5	67.9
		SOC Std (%)	8.8	28.3	17.2	25.9	29.9	29.1	27.1	28.4	29.1
	cheapest+ReOpt	Cost Avg ($)	125,259.3	31,504.4	35,613.8	49,974.0	49,528.7	31,405.1	59,809.0	31,626.9	32,008.3
		Cost Std ($)	54,481.0	3,488.4	595.8	18,412.7	16,658.7	2,718.4	21,816.3	2,575.5	2,340.0
		Cost CVaR ($)	204,123.3	37,171.2	36,521.0	79,164.2	73,993.1	35,856.7	91,683.2	35,873.6	35,929.4
		OC Cost Avg ($)	93,235.6	220.7	1.0	15,242.8	16,537.6	1.0	32,584.1	1.0	1.0
		Penalty Freq (%)	70.9	0.8	0.0	8.4	12.1	0.0	25.6	0.0	0.0
		SOC Avg (%)	34.5	65.9	91.6	65.0	63.8	64.3	65.4	64.5	66.4
		SOC Std (%)	8.9	26.8	17.5	26.8	27.8	27.6	26.4	28.3	28.4
	conservative	Cost Avg ($)	125,181.1	31,889.2	36,510.3	49,404.7	51,224.0	33,145.8	59,447.1	34,224.9	33,211.4
		Cost Std ($)	54,554.2	3,106.7	491.9	17,468.4	16,023.4	1,897.4	21,113.8	1,527.3	1,821.0
		Cost CVaR ($)	204,123.3	36,890.4	37,270.4	77,266.5	74,756.1	36,311.4	91,313.2	36,792.5	36,245.3
		OC Cost Avg ($)	93,142.6	158.4	0.1	14,771.1	17,549.9	0.1	31,747.3	0.1	0.1
		Penalty Freq (%)	72.1	0.8	0.0	8.7	12.9	0.0	25.7	0.0	0.0
		SOC Avg (%)	34.9	77.9	94.9	70.5	65.6	79.0	66.3	83.3	79.7
		SOC Std (%)	10.4	26.2	14.4	26.5	29.1	25.9	27.0	24.6	25.8
	conservative+ReOpt	Cost Avg ($)	125,082.1	31,637.1	35,707.4	48,639.4	49,046.9	31,559.6	58,676.6	31,816.6	32,118.8
		Cost Std ($)	54,692.1	3,250.4	713.3	16,917.4	16,422.3	2,717.0	21,118.5	2,551.9	2,347.4
		Cost CVaR ($)	204,123.3	36,870.8	36,884.8	75,526.3	73,555.5	35,934.5	90,419.7	35,994.2	35,996.9
		OC Cost Avg ($)	93,047.6	133.7	0.1	14,219.1	16,191.2	0.2	31,278.8	0.1	0.1
		Penalty Freq (%)	72.1	0.8	0.0	8.7	12.9	0.0	25.7	0.0	0.0
		SOC Avg (%)	35.0	81.4	94.9	73.0	66.1	79.6	67.6	77.5	77.9
		SOC Std (%)	10.5	24.2	14.3	24.4	28.4	25.2	26.2	26.9	27.2
	naïve	Cost Avg ($)	122,920.1	33,652.3	34,979.3	51,728.6	51,780.1	34,840.8	61,909.9	34,979.3	34,979.3
		Cost Std ($)	56,068.6	2,576.3	1,351.5	16,305.2	15,575.6	1,384.6	20,688.6	1,351.5	1,351.5
		Cost CVaR ($)	204,128.2	37,882.7	37,337.4	76,893.5	74,149.4	37,233.1	92,283.5	37,337.4	37,337.4
		OC Cost Avg ($)	91,216.4	186.9	1.0	14,372.6	17,575.8	1.0	31,260.6	1.0	1.0
		Penalty Freq (%)	71.7	0.7	0.0	7.9	11.1	0.0	23.7	0.0	0.0
		SOC Avg (%)	34.0	82.0	84.2	82.9	75.0	85.5	75.4	84.2	84.2
		SOC Std (%)	7.2	23.5	23.8	24.0	27.1	23.2	25.6	23.8	23.8
winter	cheapest	Cost Avg ($)	132,265.2	35,756.3	32,997.6	62,654.2	42,897.5	34,654.3	40,181.0	41,840.9	31,675.9
		Cost Std ($)	64,239.2	4,334.7	382.5	25,296.3	11,876.2	6,460.1	9,289.5	9,835.9	1,171.8
		Cost CVaR ($)	215,657.7	42,303.1	33,578.3	95,339.4	59,003.5	43,499.2	52,670.7	55,120.2	33,598.4
		OC Cost Avg ($)	111,139.4	7,027.7	0.0	32,233.9	13,672.9	5,318.1	16,950.4	18,490.6	1.6
		Penalty Freq (%)	93.0	7.2	0.0	17.1	10.1	3.3	7.0	11.3	0.0
		SOC Avg (%)	34.1	56.2	89.4	44.7	56.4	55.1	58.4	58.7	65.2
		SOC Std (%)	8.8	27.0	19.8	18.0	27.4	26.8	28.1	27.8	30.1
	cheapest+ReOpt	Cost Avg ($)	132,087.0	35,405.6	32,419.3	58,703.7	41,105.0	34,444.8	39,065.3	41,098.7	29,745.7
		Cost Std ($)	64,429.2	4,388.0	640.3	26,873.9	12,113.2	6,674.4	9,627.6	10,318.8	2,253.5
		Cost CVaR ($)	215,657.7	42,219.5	33,477.1	95,114.4	58,130.2	43,445.9	52,505.1	55,465.4	33,106.9
		OC Cost Avg ($)	110,906.4	6,705.5	0.0	28,288.4	12,632.4	5,507.2	15,631.1	17,970.9	1.6
		Penalty Freq (%)	92.7	7.0	0.0	16.5	12.0	4.0	8.2	12.5	0.0
		SOC Avg (%)	34.7	60.3	88.3	59.7	59.4	59.9	61.8	60.4	60.0
		SOC Std (%)	9.5	25.9	20.0	25.2	26.2	26.7	26.6	26.2	26.7
	conservative	Cost Avg ($)	132,038.9	35,177.8	33,159.8	59,530.0	41,745.8	34,138.1	39,218.5	40,828.1	31,802.2
		Cost Std ($)	64,453.9	3,880.3	519.4	25,427.1	11,655.2	5,903.9	8,912.2	9,403.4	1,204.6
		Cost CVaR ($)	215,657.3	41,198.2	34,014.7	93,555.5	58,392.5	42,776.1	51,993.8	54,310.1	33,725.3
		OC Cost Avg ($)	110,873.7	6,205.0	0.2	29,091.6	12,353.5	4,631.4	14,886.9	16,461.3	0.2
		Penalty Freq (%)	93.9	7.3	0.0	17.1	12.1	4.0	8.2	12.5	0.0
		SOC Avg (%)	34.8	69.6	93.3	54.5	62.1	64.8	66.4	66.3	75.4
		SOC Std (%)	10.8	27.7	16.3	23.4	27.7	27.3	28.2	27.5	26.4
	conservative+ReOpt	Cost Avg ($)	131,898.3	34,852.4	32,632.3	56,418.3	40,123.6	33,560.3	38,056.8	39,418.6	29,871.2
		Cost Std ($)	64,618.6	3,718.8	771.1	25,152.3	11,584.9	5,966.3	8,894.6	9,461.0	2,237.5
		Cost CVaR ($)	215,657.3	40,431.2	33,902.8	91,735.7	57,040.2	42,124.8	50,968.4	53,244.2	33,161.3
		OC Cost Avg ($)	110,708.5	5,977.7	0.2	25,970.7	11,474.1	4,397.6	13,880.9	15,464.9	0.2
		Penalty Freq (%)	93.9	7.3	0.0	17.1	12.1	4.0	8.2	12.5	0.0
		SOC Avg (%)	35.5	74.4	93.6	65.4	63.9	70.5	70.1	70.0	74.4
		SOC Std (%)	11.8	25.4	16.0	25.2	27.1	26.1	26.4	26.3	25.6
	naïve	Cost Avg ($)	132,574.1	37,499.6	33,394.3	57,904.2	43,047.4	37,680.1	41,547.7	42,897.5	33,514.3
		Cost Std ($)	64,059.4	2,859.5	875.8	24,685.5	10,564.7	5,178.4	8,051.1	8,416.8	875.8
		Cost CVaR ($)	215,659.6	42,050.3	34,880.6	93,695.3	58,027.3	45,129.7	52,614.1	54,524.4	34,990.1
		OC Cost Avg ($)	111,612.8	6,121.0	0.6	25,515.2	11,529.1	5,037.4	13,619.6	15,150.6	0.6
		Penalty Freq (%)	93.9	6.9	0.0	14.2	10.0	3.3	6.9	11.2	0.0
		SOC Avg (%)	31.8	72.5	87.3	74.1	77.8	86.9	83.7	83.3	87.3
		SOC Std (%)	0.2	27.0	21.8	23.5	25.6	22.6	25.2	24.9	21.8

Table 2.5 – Robust *vs.* Determinisitc RTCS performance comparison for summer and winter scenario groups. *Cost Avg* is the average *scenario cost* over all simulations. *Cost Std* represents the standard deviation of *scenario cost*. *Cost CVaR* is the conditional value at risk of scenario cost at 80% confidence level (i.e., the average scenario cost of the 20% highest scenario costs). *OC Cost Avg* is the average cost from Out of Contract (OC) energy consumption. *Penalty Freq* is the proportion of time periods with OC consumption. *SOC Avg* and *SOC Std* are the average and standard deviation of BESS State Of Charge (SOC).

The robust CCP model under budgeted uncertainty was tested with six Γ budget parameter values (0%, 20%, 40%, 60%, 80% and 100%). They indicate the proportion of the maximum allowed deviation of uncertain parameters regarding production or consumption of energy, as defined in Section 2.5.2. As a baseline for comparison, the deterministic CCP model was tested with 3 different sets of values regarding uncertain devices. When $\Phi = 0\%$, the model is based on the most optimistic scenario, assuming minimal consumption and maximal production for uncertain consumer and producer devices, respectively. The exact opposite situation is represented by $\Phi = 100\%$, apparently the most pessimistic one. Finally, the scenario where $\Phi = 50\%$ depicts the middle interval, with average values of uncertain devices. A general comparison will be conducted to determine which model behaves best under uncertainty, remembering that the Γ and Φ values used by the deterministic and robust models have different meanings.

The solution statistics for each model are presented in Table 2.3. The obtained results show that, when solving the first-stage problem to determine the list of contracts to engage, optimal solutions for robust models with different budget parameter values can be obtained in less than an hour with an 8-core CPU.

Regarding the simulation results, according to the season of the year, for each of the 4 RTCS policies proposed in the previous section, Tables 2.4 and 2.5 present statistical measures based on the operational cost (Cost Avg, Cost Std, Cost CVaR), as well as Out of Contract (OC) energy consumption cost (OC Cost Avg), penalty frequency (i.e., the proportion of time periods where OC consumption occurred) and the State of Charge (SOC) of the microgrid's BESS (SOC Avg and SOC Std). In robust and deterministic CCP models, higher values of Γ or Φ parameters, respectively, ensure improved system reliability through elevated protection against the realization of worst-case scenarios, but at the expense of increased overall cost. The above measures allow a trade-off analysis between operational cost and system reliability, which can be applied by the decision-maker to select the most appropriate model.

For the autumn season, except for the naïve RTCS policy, remark that **Rob-RTCS** solution with $\Gamma = 40\%$ significantly improves not only the operational cost (*Cost Avg* and *Cost Std*), but also microgrid's reliability (*Cost CVaR* metric), when compared to **Det-RTCS** solutions. In particular, when considering the best economic performance of **Det-RTCS** (*Cheapest+ReOpt* policy with $\Phi = 50\%$), **Rob-RTCS** with $\Gamma = 40\%$ achieves a reduction of 0.1% in *Cost Avg*, 22.7% in *Cost Std* and 3.5% in *Cost CVaR*.

In spring season instance, the look-ahead **Rob-RTCS** policies with $\Gamma = 40\%$ and $\Gamma = 80\%$ outperform the deterministic counterparts in average cost and CVaR cost metrics. When compared to the previous set of scenarios, **Rob-RTCS** performance is further improved, with $\Gamma = 40\%$ and $\Gamma = 80\%$ budget-based RTCS achieving the best economic performance (*Cost Avg*), with zero OC cost and thus zero penalty frequency. In this case, when compared to the **Det-RTCS** models whose policy has the lowest average cost (*Cheapest+ReOpt*), the **Rob-RTCS** with $\Gamma = 40\%$ is 17.6%

cheaper on average, with a 3.6% decrease in *Cost CVaR*.

For summer, the naïve policy is not able to offer improved results when coupled with the robust CCP model. On the other hand, the robust solutions with $\Gamma \in \{40\%, 80\%, 100\%\}$ provide the best protection for all look-ahead RTCS policies, when compared to their deterministic counterparts. In particular, the robust solution for $\Gamma = 40\%$ coupled with *Cheapest+ReRopt* RTCS presents the best observed economic and reliability values (*Cost Avg* = 31,405.1 and *Cost CVaR* = 35,856.7), as well as low levels of OC Cost and zero penalty frequency. It is worth noting that, besides having elevated PV electricity production, energy consumption reaches its highest levels during this season, according to the dataset.

Last, in winter season, it is possible to observe an interesting case where the maximum hedge (**Rob-RTCS** with $\Gamma = 100\%$) represents the best option from the viewpoint of worst-case protection as well as economic performance. The best results were obtained with model re-optimization enabled (either *Cheapest+ReRopt* or *Conservative+ReRopt* RTCS). The robust model with $\Gamma = 100\%$ presents the lowest *Cost CVaR* values among all models tested. It also provides the cheapest average costs (*Cost Avg*), considering all robust and deterministic models simulated. Once again, the robust-based naïve policy is not able to offer improved results when compared to **Det-RTCS**.

A statistical analysis was also performed to assess the cost difference between each pair of simulations, considering every combination of CCP model and RTCS policy. For this purpose, we applied the Wilcoxon signed-rank test [148], a non-parametric alternative to the paired Student's t-test, which does not depend on the assumption that the data is normally distributed. This test is based upon the ranks of the paired differences of measurements, and the *null hypothesis* H_0 is that two related paired samples come from the same distribution. If valid, H_0 indicates that there is no tendency for the outcome in one group of simulations to be higher or lower than in the other group. In a pair-wise comparison with a significance level $\alpha = 0.05$, considering all pairs of RTCS simulations, in only 3 cases it is not possible to reject the *null hypothesis*. These cases are related with the autumn, spring and summer instances, with no significant statistical difference when comparing the scenario costs of the *Conservative+ReOpt* RTCS based either on the **Det-RTCS** solution with $\Phi = 50\%$ or on the **Rob-RTCS** with $\Gamma = 40\%$. All other simulation comparisons yielded P-values inferior to 0.05, which indicates there is enough evidence to reject the null hypothesis and conclude that the tested samples were likely drawn from populations with differing distributions.

As a final remark, we refer the reader to the last two measures in Tables 2.4 and 2.5. According to the average and standard deviation of BESS State Of Charge, the robust-based policies rely more on the use of batteries to regulate the microgrid's system load than **Det-RTCS**.

2.7.6 Best options for CCP model and RTCS policy

The presented results confirm the overall superiority of the RTCS simulations derived from the robust CCP model results, according to the value of the budgeted uncertainty parameter Γ. Such

value will depend on the scenario type and, therefore, the season of the year. For the microgrid under study, an intermediate value of $\Gamma = 40\%$ proved to be the best parameter option for the robust model during spring, summer and autumn, while the maximum hedge ($\Gamma = 100\%$) fits best during winter season. It is worth noting that only the **Rob-RTCS** policies which incorporated the look-ahead mechanism obtained improved results when compared to the deterministic-based policies.

As a complementary evaluation, based on each season and the best performing robust CCP models presented above, we now investigate the differences among the proposed RTCS policies. Once again, we turn ourselves to Tables 2.4 and 2.5, restricting our analysis to fixed values of Γ. For each combination of season and Γ value, we split the policies into three groups: naïve, look-ahead without model re-optimization (ReOpt) and look-ahead with re-optimization applied.

As far as the robust models are concerned, it is possible to observe that the naïve policy is not able to perform well according to cost and reliability metrics, and its simulation results are inferior to those obtained by the look-ahead policies. Regarding LA policies, both *average* and *CVaR* values of scenario cost improve in re-optimization-based policies, when compared to non-re-optimized ones. As an example, considering the summer instance, re-optimized models provide an improvement of 3% in average scenario cost and 1% in *CVaR*. Out of Contract (OC) costs also decrease in most cases. One possible explanation for this behaviour is related to how the look-ahead policy works when the CCP model is re-optimized. At each time period, the LA mechanism updates the values regarding uncertain energy parameters and linear decision rules, based on a new run of the optimization model. Using these updated predictions inside the RTCS policy seems to be more cost-effective than not using them.

A second analysis, based on Pareto frontier, can also be used to determine the best-performing policy. In Figure 2.2, we plot the standard deviation of the daily cost (x-axis) versus the average of this cost (y-axis) for deterministic and robust model policies, where each point denotes a specific value of Φ or Γ, respectively. On each curve, the right most point corresponds to $\Phi = 0\%$ in the deterministic-based policies or $\Gamma = 0\%$ in the robust-based ones. Note that every point of each curve can be strictly improved in both average and std of cost by changing to a different value of Γ, without the need to trade off between average and standard deviation (std) of the cost. In other words, each point is dominated by the points to its left. Therefore, the left-most part of each curve shows the Pareto frontier of cost average *vs.* cost standard deviation performance of the associated model policy. This evaluation framework can be applied to choose a suitable value of Φ or Γ, making sure the system operates on the Pareto frontier.

For the winter instance, this means that, to retain the same level of average cost, the *robust budget model with conservative+ReOpt policy* achieves the lowest std (i.e., the highest reliability); or, conversely, to maintain the same level of std (i.e., reliability), this policy incurs the lowest average cost. That is, *robust budget / conservative+ReOpt* dominates every other policy.

Finally, in Figure 2.3, we present a graph which highlights some advantages of the robust model

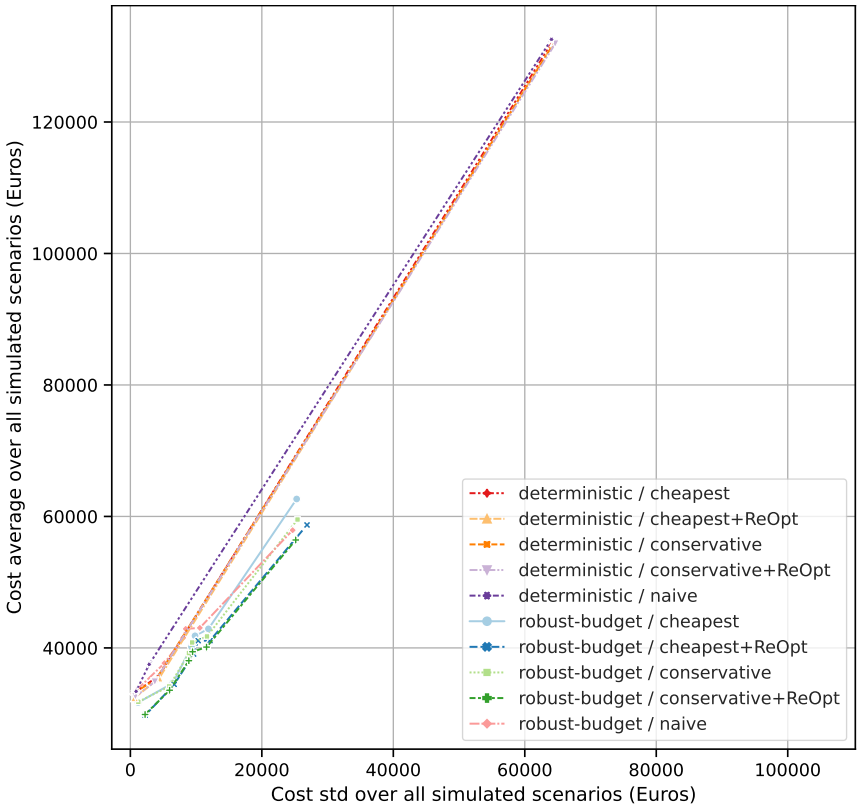

Figure 2.2 – Winter instance: daily cost std and cost average obtained with simulations of cheapest, cheapest+ReOpt, conservative, convervative+ReOpt and naïve policies, based on either deterministic or robust budget models with $\Phi = 0\%, 50\%, 100\%$ and $\Gamma = 0\%, 20\%, 40\%, 60\%, 80\%, 100\%$, respectively.

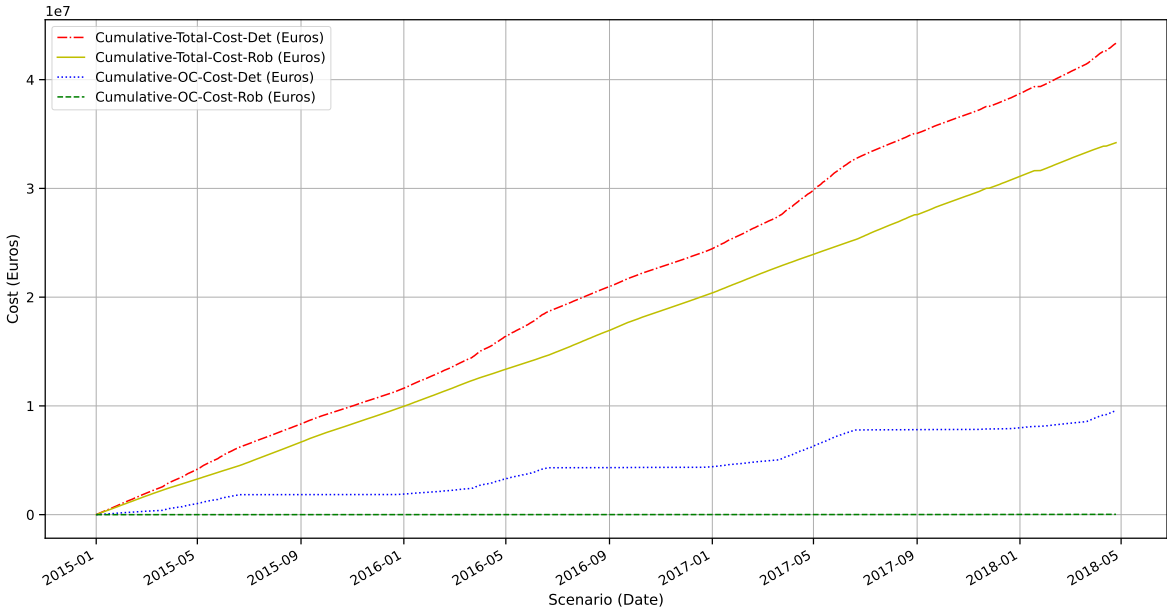

Figure 2.3 – Cumulative total costs and out-of-contract (OC) costs of the best deterministic (Det) and robust (Rob) CCP models (and associated policies) of each season, obtained after simulation over the whole time horizon (from January 2015 until May 2018).

in the long run, by comparing the accumulated costs obtained after simulating the best set of deterministic and robust CCP models and policies for each season instance, over the whole time horizon (from January 2015 until May 2018). In this analysis, at the end of the simulation time horizon, the system running with the robust model decisions incurs in no out-of-contract costs as well as significantly cheaper accumulated total cost, 21% less when compared to the best deterministic model.

2.8 Discussion

This chapter presented the Contract Collaboration Problem (CCP), a multi-contract energy trading framework based on flexible commitments, coupled with a Real-Time Command Strategy (RTCS) for usage in microgrid energy trading and scheduling. As the main component, we developed a robust model under budgeted uncertainty which provides protection against the worst-case realization of the microgrid's production and consumption of electricity, by presenting a cost-effective contract commitment planning for a given time horizon.

A case study was conducted on a real microgrid, with a total of four problem instances, one for each season of the year. Monte-Carlo simulations were used to assess the performance of the proposed CCP robust model solution (against the deterministic alternative), when used as input for real-time energy scheduling strategies. Relying on a set of real-world-inspired energy purchase contracts, simulation results have confirmed the efficacy of different robust-based RTCS strategies,

according to scenario types. For specific protection levels, the robust RTCS was able to dominate the deterministic RTCS in all operational cost and system reliability metrics.

Overall, the CCP robust model under budgeted uncertainty provides a pool of solutions with different protection levels so that decision-makers can pick according to their preferences. The effectiveness of each robust solution will depend on the microgrid's load profile and renewable production, which vary according to the season of the year.

Chapter 3

The Robust Permutation Flow Shop Problem (makespan objective)

In permutation flow shop scheduling problems, the operations concerning each job are performed in a serial flow (i.e., in a specific production sequence) on each machine, and the processing order of the jobs is the same for each subsequent step. Such configuration is commonly used in assembly lines throughout several types of industry, including chemical [22], petrochemical [30], automobile manufacturing [155], electronic [64], food and metallurgical [52]. In all these applications, maintaining a continuous flow of processing tasks while minimizing idle time and waiting time is highly important, guaranteeing process efficiency and increasing production rates and profits.

The Permutation Flow Shop (PFS) scheduling has been widely investigated considering deterministic processing times, assuming input data are accurate and known in advance. However, in real-world industrial settings, manufacturing systems usually operate in uncertain environments where interruptions (essentially random in nature) restrict the execution of production schedules precisely as they were planned. In particular, variation in processing times and other stochastic events (e.g., machine breakdowns, due-date changes, order cancellations, and raw material shortages) bring increased variability in production systems and may influence the optimization model's quality and feasibility [119].

A typical solution approach applied to these cases is Stochastic Optimization [53], whose fundamental premise states that the unknown parameters can be described using probabilities. On the other hand, Robust Optimization (RO) [14] provides a different approach to optimization problems under uncertain conditions. The unknown parameters are modeled as belonging to an uncertainty set, making it a more suitable modeling approach when the uncertainty interval is known, but not necessarily the probability distribution. The goal for RO is to obtain a feasible decision, no matter what the constraints turn out to be, which is optimal for the worst-case objective function.

The objective of this chapter is to provide solution methods for the Robust Permutation Flow Shop to minimize the worst-case *makespan*. Assuming processing times are uncertain and vary in a given interval, the only information required is the lower and upper bounds of processing times, obtained from historical data. We are interested in a job permutation that minimizes the worst-case *makespan* for any possible realization of job processing times under the budgeted uncertainty

set [16]. Unlike other robust optimization models, which generate only one conservative solution, the budgeted approach allows the adjustment of the level of conservatism of the solution, allowing the incorporation of different attitudes toward risk (e.g., risk-averse, risk-neutral, or risk-seeking). As a result, the decision-maker can select the schedule that achieves the best balance between robustness and optimality.

3.1 Overview

The first problem to be analyzed is the 2-machine Robust Permutation Flow Shop (2RPFS) problem to minimize the worst-case makespan. The objective is to provide exact solutions to this variation of the problem, by developing an exact method, based on the Column-and-Constraint (C&CG) generation algorithm. Afterwards, we solve the m-machine version of the problem, with the application of both metaheuristics and exact methods.

We start this chapter introducing the classical Permutation Flow Shop (PFS) Problem, whose processing times are deterministic (i.e. fixed). The four most well-known MILP models for this problem are also presented, in Section 3.2.3. An extensive literature review on solution methods for the PFS problem with uncertain processing times is presented in Section 3.3. The two-machine Robust Permutation Flow Shop Problem is introduced in Section 3.4, together with the budgeted uncertainty set and two proposed robust counterpart formulations. In sequence, Section 3.5 describes an exact solution method to the 2RPFS problem, which uses a dynamic-programming-based worst-case procedure. The obtained performance is then discussed in Section 3.6, based on extensive computational experiments on existing literature instances.

The second part of this chapter is devoted to the m-machine version of the Robust Permutation Flow Shop (RPFS) problem and proposes, in Section 3.7, an extension of the 2RPFS budget uncertainty set, now considering m machines and a single budget parameter. This section also presents extended versions of the Robust Counterparts originally introduced in Section 3.4.3, which can now be used to solve the RPFS problem with an arbitrary number of machines. A baseline exact solution method for the problem is then depicted in Section 3.8, generalizing the solution method and worst-case procedures initially proposed in Section 3.5. With solution method efficiency in mind, Section 3.9 introduces a Greedy Randomized Adaptive Search Procedure (GRASP) metaheuristic to solve the RPFS, followed by Section 3.10, with the obtained results of exact and heuristic solution methods. Finally, we close this chapter with a general discussion on the results obtained when solving both 2RPFS and m-machine RPFS problems, in Section 3.11.

3.2 The Deterministic Permutation Flow Shop (PFS) Problem

This section presents the Permutation Flow Shop Problem (PFSP) to minimize the *makespan*. Following the well-known $\alpha|\beta|\gamma$ notation for scheduling problems, established by [50], where α represents the machine environment, β stands for job characteristics, and γ symbolizes the objective function, this problem is denoted as $F|prmu|C_{max}$. Since job processing time values are assumed to be known in advance, we will use the term deterministic when referring to this version of the problem.

3.2.1 Problem definition

The problem can be stated as follows. Consider a production planning process consisting of a set $\mathbb{J} = \{J_1, J_2, \ldots, J_n\}$ of n jobs to be executed in a set $\mathbb{M} = \{M_1, M_2, \ldots, M_m\}$ of m machines. In this process, every job J_i is composed of m stages $O_{1,i}, O_{2,i}, \ldots, O_{m,i}$, named operations. Each job operation $O_{r,i}$ must only be executed on machine M_r. Every operation $O_{r,i}$ has a non-negative processing time $p_{r,i}$ forming the matrix $P \in \mathbb{R}^{+}_{\mathbb{M}\times\mathbb{J}}$. At any time, each machine cannot execute more than one operation. Operation $O_{r,i}$ can only be executed after operation $O_{r-1,i}$ is finished. Preemption is not allowed: once an operation is started, it must be completed without any interruption. The permutation flow shop's particularity is that the sequence in which the jobs are processed (permutation) is the same for all machines. Such sequence is defined by a permutation σ: $\{1, \ldots, n\} \longrightarrow \mathbb{J}$, with $\sigma(j)$ indicating the jth job to be executed. We call Σ the set of all permutations of n jobs, hence $\sigma \in \Sigma$. The completion time of an operation $O_{r,\sigma(j)}$, denoted by $C_{r,\sigma(j)}$, can be defined by the recurrence:

$$C_{r,\sigma(j)} = \begin{cases} p_{r,\sigma(j)} & \text{if } r = 1 \text{ and } j = 1, \\ C_{r,\sigma(j-1)} + p_{r,\sigma(j)} & \text{if } r = 1 \text{ and } j > 1, \\ C_{r-1,\sigma(j)} + p_{r,\sigma(j)} & \text{if } r > 1 \text{ and } j = 1, \\ max(C_{r,\sigma(j-1)},\ C_{r-1,\sigma(j)}) + p_{r,\sigma(j)} & \text{if } r > 1 \text{ and } j > 1. \end{cases}$$

The completion time of a job J_i is defined as its completion time on the last machine: $C_{m,i}$. The *makespan* is the maximum completion time C_{max}, considering all jobs, i.e., $C_{max} = \max_{i\in\{1,\ldots n\}} C_{m,i} = C_{m,\sigma(n)}$. The PFSP objective is to find a sequence of jobs (permutation σ) that minimizes the *makespan*.

As far as the *makespan* objective is concerned, the PFSP was proved strongly NP-hard by [44] for instances with three or more machines. However, the two-machine version of the problem can be solved in $O(n \log n)$ time with the well-known algorithm proposed by [66] in one of the pioneering papers in the scheduling literature. Indeed, for the two-machine case, the optimal *makespan* for the deterministic flow shop problem is the same with or without the permutation constraint (i.e., the sequence in which the jobs are processed is the same for all machines) [109].

Several solution approaches have been developed for the PFS problem with makespan minimization: Mixed Integer Linear Programming (MIP) models [136], approximation algorithms [101, 110], *branch-and-bound* [111, 21, 75], *Simulated Annealing* [62], *Tabu Search* [137] and Genetic Algorithms [154], to name a few.

To our knowledge, no proven optimal solutions for the problem are currently available for instances with more than 2000 operations (n = 200, m = 10). Therefore there is scope for new approaches that provide either exact or approximate improved solutions for larger instances.

3.2.2 Nomenclature used in the models

Indices and sets

n number of jobs
m number of machines
r machine indices; $r \in \{1, 2, \ldots, m\}$
i, k job indices; $i, k \in \{1, 2, \ldots, n\}$
j sequence position; $j \in \{1, 2, \ldots, n\}$
$\mathbb{J}$ set of job indices: $\mathbb{J} = \{1, 2, \ldots, n\}$
$\mathbb{M}$ set of machine indices: $\mathbb{M} = \{1, 2, \ldots, m\}$

Input parameters

$\{p_{ri}\}$ $m \times n$ matrix of processing times of job i on machine r

Model variables

B_{rj} start (begin) time of job in sequence position j on machine r
E_{rj} completion (end) time of job in sequence position j on machine r
S_{ri} start time of job i on machine r
C_{ri} completion time of job i on machine r
C_{max} maximum flowtime (makespan) of the schedule determined by the completion time of the job in the last sequence position on the last machine
X_{rj} idle time on machine r before the start of job in sequence position j
Y_{rj} idle time of job in sequence position j after it finishes processing on machine r

$$D_{ik} = \begin{cases} 1, & \text{if job } i \text{ is scheduled any time before job } k \\ 0, & \text{otherwise} \end{cases}$$

$$Z_{ij} = \begin{cases} 1, & \text{if job } i \text{ is assigned to sequence position } j \\ 0, & \text{otherwise} \end{cases}$$

D_{ik} and Z_{ij} are binary integer variables, while the others are real variables that take integer values when processing times are also integer values. The four most popular MILP models for the PFS Problem are described next.

3.2.3 Mixed Integer Programming Formulations

Several Mixed-integer linear programming (MILP) models have been developed for the PFSP with an arbitrary number of machines. For a comparative analysis among the best performing PFSP formulations, we refer the reader to [135], which presents an empirical study based on a standard set of 60 problem instances. We now present four competing MILP models designed for the PFS Problem.

3.2.3.1 The Wagner model

Wagner [142] formulated an all-integer programming model for a three-machine flow shop, which was later extended to problems where $m > 3$ by Baker [8] and Stafford [129]. The all-integer model was converted to an MILP model by Stafford. The following is a corrected and simplified version of this model:

$$\text{minimize } C_{max} = C_{mn} \tag{3.1}$$

$$\text{subject to } \sum_{j \in \mathbb{J}} Z_{ij} = 1 \qquad \forall\, i \in \mathbb{J} \tag{3.2}$$

$$\sum_{i \in \mathbb{J}} Z_{ij} = 1 \qquad \forall\, j \in \mathbb{J} \tag{3.3}$$

$$\sum_{i \in \mathbb{J}} p_{ri} Z_{i,j+1} - \sum_{i \in \mathbb{J}} p_{r+1,i} Z_{ij} + X_{r,j+1} - X_{r+1,j+1} + Y_{r,j+1} - Y_{rj} = 0; \quad (1 \leq r \leq m-1; 1 \leq j \leq n-1) \tag{3.4}$$

$$\sum_{i \in \mathbb{J}} p_{ri} Z_{i1} + X_{r1} - X_{r+1,1} + Y_{r1} = 0; \quad (1 \leq r \leq m-1) \tag{3.5}$$

$$C_{mn} = \sum_{i \in \mathbb{J}} p_{mi} + \sum_{p \in \mathbb{J}} X_{mp} \tag{3.6}$$

Constraints (3.2) and (3.3) are the classical assignment problem, ensuring that each job is assigned to one, and only one sequence position; and that each sequence position is filled by one, and only one job. Constraints (3.4) and (3.5), called Job-Adjacency Machine-Linkage (JAML) constraints, ensure that (a) the job in sequence position j cannot begin processing on machine $r+1$ until it has completed its processing on machine r; and (b) the job in sequence position $j+1$ cannot begin its processing on machine r until the job in sequence position j has completed its processing on that same machine. Constraint (3.5) was not included in Wagner's work, but was later proved to be necessary by Stafford and Tseng [130]. Constraint (3.6) measures the makespan of the set of jobs.

3.2.3.2 The Wilson model

Alternatively, Wilson [149] applied sets of inequality constraints to control the JAML relationships, similar to those used in the Manne models, described in the next section. His model is formulated as follows:

$$\text{minimize } C_{max} = B_{mn} + \sum_{i \in \mathbb{J}} p_{mi} Z_{in} \tag{3.7}$$

$$\text{subject to } \sum_{j \in \mathbb{J}} Z_{ij} = 1 \qquad \forall\, i \in \mathbb{J} \tag{3.8}$$

$$\sum_{i \in \mathbb{J}} Z_{ij} = 1 \qquad \forall\, j \in \mathbb{J} \tag{3.9}$$

$$B_{1j} + \sum_{i \in \mathbb{J}} p_{1i} Z_{ij} = B_{1,j+1} \;\; (1 \leq j \leq n-1) \tag{3.10}$$

$$B_{11} = 0 \tag{3.11}$$

$$B_{r1} + \sum_{i \in \mathbb{J}} p_{ri} Z_{i1} = B_{r+1,1}; \;\; (1 \leq r \leq m-1) \tag{3.12}$$

$$B_{rj} + \sum_{i \in \mathbb{J}} p_{ri} Z_{ij} \leq B_{r+1,j}; \;\; (1 \leq r \leq m-1; 2 \leq j \leq n) \tag{3.13}$$

$$B_{rj} + \sum_{i \in \mathbb{J}} p_{ri} Z_{ij} \leq B_{r,j+1}; \;\; (2 \leq r \leq m; 1 \leq j \leq n-1) \tag{3.14}$$

The assignment problem (previously described in Wagner's model) is represented by constraints (3.8) and (3.9). Constraints (3.10), (3.11), and (3.12) guarantee no idle time on machine 1, and that job 1 is processed by all m machines without delay. Constraints (3.13) assure that the start of each job on machine $r+1$ is no earlier than its finish on machine r. Constraints (3.14) ensure that the job in position $j+1$ in the sequence only starts on machine r after the job in position j in the sequence has completed its processing on that machine.

3.2.3.3 The Manne model

Following a different approach, Manne [94] initially proposed a dichotomous-constraints integer programming model for the general job shop problem, which assures that only one of each pair of constraints can hold: job i either precedes job k somewhere in the processing sequence, or job k precedes job i. The original Manne model was later adapted to the PFS by Stafford and Tseng [67]:

$$\text{minimize } C_{max} \tag{3.15}$$

$$\text{subject to } C_{1i} \geq p_{1i}; (i \in \mathbb{J}) \tag{3.16}$$

$$C_{ri} - C_{r-1,i} \geq p_{ri}; \;\; (2 \leq r \leq m; i \in \mathbb{J}) \tag{3.17}$$

$$C_{ri} - C_{rk} + PD_{ik} \geq p_{ri}; \;\; (r \in \mathbb{M}; 1 \leq i < k \leq n) \tag{3.18}$$

$$C_{rk} - C_{ri} + P(1 - D_{ik}) \geq p_{rk}; \;\; (r \in \mathbb{M}; 1 \leq i < k \leq n) \tag{3.19}$$

$$C_{max} \geq C_{mi}; \;\; (i \in \mathbb{J}) \tag{3.20}$$

Constraints (3.16) assure that the completion time of each job on machine 1 must be at least the duration of that job's processing time on the first machine. Constraints (3.17) ensure that each job's

completion time on machine r is no earlier than that job's completion time on machine $r-1$ plus that job's processing time on machine r. With P defined as a very large constant, the paired disjunctive constraints (3.18) and (3.19) make sure that job k is either ahead of job i or comes after job i in the sequence, but not both. Constraint (3.20) defines the *makespan* as the maximum completion time of all jobs on the last machine.

3.2.3.4 The Liao–You model

In Liao and You [89], an alternative Job Shop model was proposed. Each pair of inequality dichotomous constraints from the original Manne [94] model was algebraically combined into a single equality constraint, which equals a surplus variable q_{rik}. Such variable controls the precedence relationship of jobs i and k on machine r. The model also assumes P is a very large constant. Finally, in order to guarantee feasibility, the authors added a boundary constraint on the surplus variables.

Pan [107] later developed an adaptation of this model to the Permutation Flow Shop Problem, which is stated below.

$$\text{minimize } C_{max} \tag{3.21}$$

$$\text{subject to } S_{ri} + p_{ri} \leq S_{r+1,i}; \quad (1 \leq r \leq m-1; i \in \mathbb{J}) \tag{3.22}$$

$$S_{ri} - S_{rk} + PD_{ik} - p_{rk} = q_{rik}; \quad (r \in \mathbb{M}; 1 \leq i < k \leq n) \tag{3.23}$$

$$q_{rik} \leq P - p_{ri} - p_{rk}; \quad (r \in \mathbb{M}; 1 \leq i < k \leq n) \tag{3.24}$$

$$C_{max} \geq S_{mi} + p_{mi}; \quad (i \in \mathbb{J}) \tag{3.25}$$

Where constraints (3.22) represent a lower bound on the start time of job i on the subsequent machine $r+1$, constraints (3.23) are the algebraic combination of the paired disjunctive constraints (3.18) and (3.19) from the Manne model, and constraints (3.24) define an upper bound on the respective surplus variable q_{rik}. Finally, constraints (3.25) represent the *makespan* value (largest completion time on the last machine m, given all jobs $i \in \mathbb{J}$, such that $C_{mi} = S_{mi} + p_{mi}$).

3.3 Literature review

This section provides an overview of the flow shop problem with uncertain processing times, concerning existing Stochastic Optimization methods (Section 3.3.1) and Robust Optimization approaches (Section 3.3.2).

3.3.1 Stochastic Optimization

[48] surveyed 100 papers that study uncertainty in different variations of flow shop scheduling problems, published between 2001 and 2016. According to their analysis, the two most common

uncertain parameters are processing times and machine breakdowns. The majority of works are related to Stochastic Optimization, including heuristics [31, 34, 9, 42], simheuristics [40], probabilistic hybrid heuristics [77], branch-and-bound [10], and simulation [43].

It is worth noting that Stochastic Optimization approaches model random parameters using probability distributions, which may be difficult to infer in many cases. Additionally, optimizing the expected value of an objective may not be the best alternative for processes that incorporate only a small number of trials. The benefits of optimum expected value shall only be visible in the long haul, after a large number of tests.

The problem was also investigated from the viewpoint of fuzzy programming [55, 95] and stability analysis [78, 20].

3.3.2 Robust Optimization

When applying Robust Optimization techniques, no assumptions are necessary concerning the underlying probability distribution of uncertain data. Also, different approaches towards risk can be incorporated into the problem. As far as robust scheduling problems are concerned, the objective is to optimize a performance measure considering the worst possible scenario, thus developing a schedule that will perform relatively well under a wide range of possible realizations of processing times. Different optimization criteria may be used to choose a robust solution [2]. The first and most straightforward criterion is the *minimax* (also known as the *absolute robust* criterion). In this case, given a minimization problem, the robust decision is made by choosing a solution that minimizes the highest cost over all possible scenarios.

A second possible criterion is minimax regret, aiming to find the least maximum regret over all possible scenarios. Regret can be either defined as the *difference* or the *ratio* between the resulting cost of the candidate decision and the cost of the decision that would have been taken if uncertain input data were known in advance (before the decision time, i.e., before solving the problem). In the first case, where regret is defined as a difference, the so-called *robust deviation* decision is obtained. For the latter case (regret being the ratio of two values), the resulting decision is the *relative robust* decision.

Concerning the uncertain nature of the problem, scenarios represent the set of possible realizations of processing time values. When applying RO, there are two usual ways of describing the set of scenarios. In the discrete case, an explicit scenario list is given, i.e., one processing time matrix P^{λ} for each scenario λ. In the interval case, for each operation $O_{r,i}$ concerning the execution of job i on machine r, a range $[p^{L}_{r,i},\ p^{U}_{r,i}]$ of lower and upper bounds of processing times is defined. The so-called continuous processing time interval involves, therefore, an infinite number of scenarios. Regardless of the scenario representation approach, interval or discrete, once the robust counterpart formulations have been defined, developing an appropriate solution method remains a challenge.

Table 3.1 summarizes existing works regarding the makespan-objective Robust Permutation Flow Shop Problem, in terms of optimization criterium (problem type), solution approach (heuristics or exact methods), the number of machines, and how processing time uncertainty was represented (discrete or interval).

Given the minimax regret makespan criterion, [74] studied both the discrete and interval cases of processing time uncertainty. Their work proposed branch-and-bound and heuristic procedures to obtain robust solutions, along with experiments based on a considerable set of randomly generated instances, with no more than 15 jobs, however. They also provided NP-hardness proof for the discrete scenario case of the robust flow shop with $m = 2$. Very recently, the 2-machine interval processing time case has been proven to be NP-hard by [120].

Problem Type	Heuristics / Approximation	Exact methods
Minimax Regret	**2 machines**: Greedy [74]$^{\mathrm{D,\ I}}$ **3 machines**: Evolutionary [28]$^{\mathrm{I}}$ **m machines**: Constructive [29]$^{\mathrm{I}}$ Scatter Search [123]$^{\mathrm{I}}$	**2 machines**: Branch & Bound [74]$^{\mathrm{D,\ I}}$ **2 jobs**: $O(m)$ [6]$^{\mathrm{I}}$
Minimax	**2 machines**: PTAS [71]$^{\mathrm{D}}$	-
Minimax, Budgeted uncertainty	**2 machines**: SA and IG [159]$^{\mathrm{I}}$	-

Table 3.1 – Summary of algorithms listed in the literature review regarding the Robust PFSP (*makespan* objective). For each work, we specify how processing time uncertainty was represented: a **D** means discrete processing time matrices; an **I** means processing time intervals.

Concerning the existing solution approaches involving only discrete-scenario uncertainty, [71] studied the two-machine permutation flow shop problem ($F_2 \mid prmu \mid C_{max}$) with uncertain data, whose deterministic counterpart is known to be polynomially solvable. The work proved that the minimax and minimax regret versions of the problem are strongly NP-hard even for two scenarios. The authors developed a polynomial-time approximation scheme (PTAS) algorithm to be used with the minimax version of the problem when the number of scenarios is constant. They also stated that the minimax regret version is not at all approximable, even for two scenarios.

Regarding the interval scenario case with the minimax makespan criterion, the computational complexity of the robust flow shop with $m = 2$ is still an unsolved open problem [71]. Note that the problem is NP-hard for $m \geq 3$ following the complexity of the deterministic problem. Several solution strategies deal exclusively with processing time uncertainty represented as intervals of known bounds. [6] studied the minimax regret flow shop with m machines but only two jobs. [123] proposed a Scatter Search metaheuristic for the PFSP and other two scheduling problems with

interval uncertainty, based on the minimax regret criterion.

Unlike previous works, [159] adopted a new approach, which searches for a minimax robust schedule given the restricted worst-case scenario, based on the [16] budgeted uncertainty set. Simulated annealing (SA) and Iterated Greedy (IG) metaheuristics were applied to solve the problem over a set of 300 randomly generated instances.

In the same year, [28] proposed an evolutionary algorithm for the minimax regret makespan robust flow shop with three machines, assuming processing times belonged to known intervals. Later, the same authors proposed another solution approach for an arbitrary number of machines [29], where a constructive algorithm based on the Nawaz-Enscore-Ham (NEH) heuristic [103] has been introduced and experimentally evaluated against two other heuristic algorithms: the authors' evolutionary algorithm and a Middle Interval heuristic.

3.4 The two-machine Robust Flow Shop Problem (2RPFS)

From this moment on, we will concentrate on the two-machine robust flow shop problem. Different optimization criteria may be used to choose a robust solution [46, 2]. Our research focuses on the *minimax* criterion, also known as the *absolute robust* criterion. In this case, considering a minimization problem, the robust decision is made by choosing a solution that minimizes the highest solution value over all possible scenarios, according to a predefined uncertainty set. We refer the reader to Appendix A for a summary of the two types of optimization criteria that can be applied when solving the robust version of the problem.

Given that the computational complexity remains an open problem for 2RPFS under budgeted uncertainty, in this work, we fill a gap in the literature by providing an exact solution method (see Table 3.1 from Section 3.3). Furthermore, it is worth noting that the models and the solution method presented in this section can be generalized to the m-machine variant of the robust PFSP, which is NP-hard for $m \geq 3$, following the complexity of the deterministic problem.

After defining the 2RPFS problem, this section describes the application of the budgeted uncertainty set. Finally, two robust counterpart formulations are proposed, based on well-known Mixed-Integer Linear Programming (MILP) formulations for the deterministic problem.

3.4.1 Problem statement

Assume the matrix of individual processing times $\mathbf{P} = \{p_{r,i}\}$ as uncertain. A scenario λ is defined as a realization of uncertainty and, for each λ, there is a unique matrix of processing times $\mathbf{P}^{\lambda} = \{p^{\lambda}_{r,i} \in \mathbb{R} : r = 1, 2, i = 1, 2, \ldots, n\}$. We define Λ as the set indexing all possible scenarios.

Let $\varphi(\sigma, \mathbf{P}^{\lambda})$ be the makespan of a sequence $\sigma \in \Sigma$ in scenario λ (i.e., given processing time matrix $\mathbf{P}^{\lambda}$). The objective of the two-machine robust (minimax makespan) flow shop is to find a job

permutation $\sigma \in \Sigma$ that *minimizes* the *maximum* possible **makespan** over all possible scenarios $\lambda \in \Lambda$:

$$\textbf{2RPFS:} \quad \min_{\sigma \in \Sigma} \max_{\lambda \in \Lambda} \{\varphi(\sigma, \mathbf{P}^{\lambda})\} \tag{3.26}$$

For any sequence $\sigma \in \Sigma$, the value

$$\mathcal{Z}(\sigma) := \max_{\lambda \in \Lambda} \{\varphi(\sigma, \mathbf{P}^{\lambda})\} \tag{3.27}$$

is called the *worst-case makespan* or *robust cost* for σ. A maximizer in (3.27) is called a worst-case scenario for σ.

3.4.2 Budgeted uncertainty set for the 2RPFS problem

[159] compared the three classical Robust-Counterpart Optimization (RCO) models in terms of the number of variables, the number of required constraints, and if the respective formulation is linear or not. The first and simplest model, by [128], consists of a linear formulation that presents the smallest number of variables and constraints. However, it is not possible to adjust its degree of solution conservatism. Following a new approach, [15] proposed an RCO model with safety parameters, which allow a trade-off between robustness and performance. However, the resulting formulation is non-linear (conic quadratic) and more challenging to solve than the original problem. Finally, the model developed by [16] provided a linear formulation that allows controlling the level of conservatism of the robust solution without resulting in a substantial increase in problem size. With the inclusion of a budget parameter for each constraint, it is possible to adjust the number of coefficients that simultaneously take their largest variations, thus providing a compromise between robustness and optimality. Therefore, the so-called budgeted uncertainty set, defined by this RCO model, comes as a natural choice to model processing time uncertainty in scheduling problems.

The budgeted uncertainty set for the 2RPFS problem was proposed by [159]. We reproduce their definition below, with some modifications in notation as well as additional comments.

As stated in Section 3.3, there are two common ways of representing scenario set Λ. Given the interval approach for representing uncertain values, consider two positive processing time matrices $\overline{\mathbf{P}}$=$\{\overline{p}_{r,i}\}$ and $\widehat{\mathbf{P}}$=$\{\widehat{p}_{r,i}\}$, that represent the nominal values of and the maximum allowed deviations of $\mathbf{P}$, respectively. Additionally, we introduce two positive integers Γ_1 and Γ_2, which will be called budget parameters, that denote the maximum number of operations whose uncertain processing times can reach their worst-case values in machines M_1 and M_2, respectively. We can define the bounded (processing time) uncertainty sets of operations in M_1 and M_2, denoted as $\mathcal{U}_r$ $(r = 1, 2)$, as follows:

$$\mathcal{U}_r = \left\{ \mathbf{P}_r = \{p_{r,i}\} : p_{r,i} = \overline{p}_{r,i} + \delta_{r,i}\, \widehat{p}_{r,i},\ \delta_{r,i} \in \{0,1\}, i \in \{1, \ldots, n\},\ \sum_{i=1}^{n} \delta_{r,i} \leq \Gamma_r \right\}, \tag{3.28}$$

where $\mathbf{P}_r$ is the projection of $\mathbf{P}$ in the space defined by machine M_r $(r = 1, 2)$.

We can now define the *budgeted uncertainty set* as:

$$\mathcal{U}_\Gamma = \mathcal{U}_{(\Gamma_1,\Gamma_2)} = \mathcal{U}_1 \times \mathcal{U}_2.$$

Notice that a scenario $\lambda \in \Lambda$ is defined by the matrix $P^\lambda \in \mathcal{U}_\Gamma$. Also, for a given $r \in \mathbb{M}$ and $i \in \mathbb{J}$, let $\delta^\lambda_{r,i}$ be the value defining the deviation of the processing time regarding the execution of job i on machine r in scenario λ, i.e., $p^\lambda_{r,i} = \overline{p}_{r,i} + \delta^\lambda_{r,i}\widehat{p}^\lambda_{r,i}$. Therefore, the total number of operations whose processing time can deviate to its maximum value in machine M_r is limited to Γ_r.

The main advantage of applying budgeted uncertainty sets is the ability to model the risk-averseness of the decision-maker by varying Γ_1 and Γ_2. As mentioned by [16], the idea is that an event where all uncertain parameters $p_{r,i}$ reach their worst-case values at the same time has a very low probability of happening. In particular, higher values of Γ_1 and Γ_2 lead to larger uncertainty sets and thus more conservative solutions. When $\Gamma_1 = \Gamma_2 = 0$, the problem is equivalent to the nominal problem, i.e., the PFSP with two machines. If $\Gamma_1 = \Gamma_2 = n$, we obtain the box uncertainty set [128]. For a given value of Γ_1 and Γ_2, there are $\binom{\Gamma_1}{n} \times \binom{\Gamma_2}{n}$ possible worst-case scenarios, given the budgeted uncertainty set $\mathcal{U}_\Gamma$.

[159] affirmed that obtaining an exact solution for the two-machine Robust PFSP under budgeted uncertainty would be computationally intractable for real-sized problem instances. However, with the method introduced in this research, it turns out that it is possible to obtain exact solutions for most instances from that work in reasonable execution time (see results in Section 3.6).

3.4.3 Robust counterparts

We now present the robust counterparts for Wagner [142] and Wilson [149] PFSP MILP models. According to the empirical study conducted by [135], these two assignment-problem-based models are the best performing ones, based on results obtained on a standard set of 60 problem instances. In both models, the number of constraints and the model matrix size are smaller than the other two competing integer programming models from the literature [67, 89]. Experimental data suggests that this factor greatly influences the computational time of the PFS models, apparently more than the number of binary variables.

3.4.3.1 Robust Counterpart for Wagner PFS Model

[142] proposed an all-integer programming model for a three-machine deterministic flow shop, later extended to an m-machine MILP model by [129], and commonly named in the literature as *wagner model*. We now present its robust counterpart for two machines. In this two-stage RO formulation, y and $Z_{i,j}$ are the first-stage variables, while X^λ_j and Y^λ_j are in the second stage.

$Z_{i,j} = \begin{cases} 1, \text{ if } \sigma(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma) \\ 0, \text{ otherwise.} \end{cases}$	
X_j^λ	idle time on machine M_2 before the start of operation concerning the job in sequence position j given scenario λ.
Y_j^λ	idle time of the job in sequence position j after it finishes processing on machine M_1 given scenario λ.

$$\min\ y \tag{3.29}$$

$$\text{st } \sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta_{2,i}^{\lambda}\right) + \sum_{j=1}^{n} X_j^\lambda \leq y, \qquad \lambda \in \Lambda, \tag{3.30}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^{\lambda}\right) Z_{i,j+1} + Y_{j+1}^\lambda = \sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta_{2,i}^{\lambda}\right) Z_{i,j} + X_{j+1}^\lambda + Y_j^\lambda,$$

$$1 \leq j \leq n-1, \lambda \in \Lambda, \tag{3.31}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^{\lambda}\right) Z_{i,1} = X_1^\lambda, \qquad \lambda \in \Lambda, \tag{3.32}$$

$$\sum_{i=1}^{n} Z_{i,j} = 1, \qquad j = 1, \ldots, n, \tag{3.33}$$

$$\sum_{j=1}^{n} Z_{i,j} = 1, \qquad i = 1, \ldots, n, \tag{3.34}$$

$$Z_{i,j} \in \{0,1\}, \qquad i, j = 1, \ldots, n, \tag{3.35}$$

$$X_j^\lambda \geq 0,\ Y_1^\lambda = 0,\ Y_j^\lambda \geq 0, \qquad j = 1, \ldots, n, \lambda \in \Lambda, \tag{3.36}$$

$$y \geq 0. \tag{3.37}$$

The objective function (3.29) and constraint (3.30) have the goal of finding a robust schedule that minimizes the *makespan* y of the worst-case scenario, among all possible scenarios. Constraints (3.31) and (3.32) are the Job-Adjacency and Machine-Linkage (JAML) constraints from the Wagner model, written for each scenario $\lambda \in \Lambda$. We refer the reader to Figure 3.1 for an illustrative JAML diagram. They ensure that, for each scenario λ: (a) the job in sequence position j cannot begin processing on machine M_2 until it has completed its processing on machine M_1, and (b) the job in sequence position $j+1$ cannot begin its processing on machine M_r until the job in sequence position j has completed its processing on that same machine. Remark that the original Wagner model does not enforce an important aspect: all jobs are processed on machine M_1 without any in-sequence machine idleness, i.e., the idle time before any job is processed on machine M_1 is always zero. As a consequence, the idle time of the first job after processing on M_1 is zero ($Y_1 = 0$).Constraints (3.33)

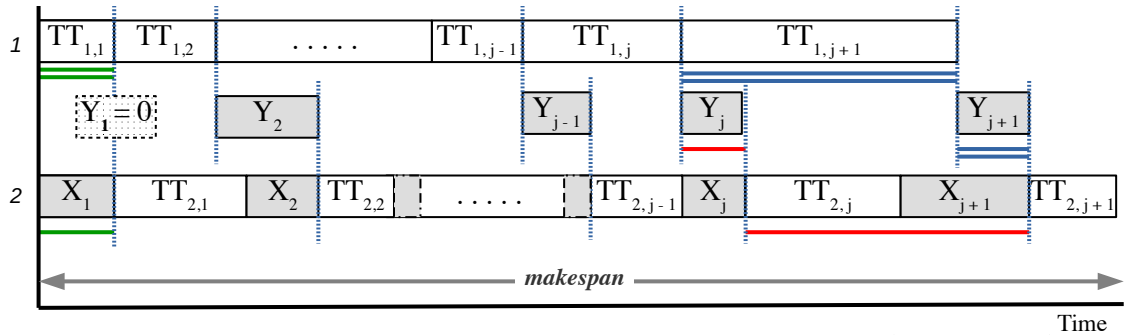

Figure 3.1 – JAML diagram for Wagner model, where $TT_{r,j} = \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^{\lambda}\right) Z_{i,j}$.

and (3.34) are the classical assignment constraints, ensuring, respectively, that each job is assigned to one and only one sequence position; and that each sequence position is filled by one and only one job. Finally, constraints (3.35)-(3.37) define the domain of the variables.

3.4.3.2 Robust Counterpart for Wilson PFS Model

Rather than using equality constraints and idle time variables for controlling the so-called JAML relationships, [149] applied sets of inequality constraints based on start time variables defined for each job operation and each machine. This variant of the model uses the following decision variables:

$Z_{i,j} = \begin{cases} 1, \text{ if } \sigma(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma) \\ 0, \text{ otherwise.} \end{cases}$
$B_j^\lambda =$ start time of operation concerning job $\sigma(j)$ (in position j) on machine M_2 given scenario λ.

Based on the above definitions, where variables $Z_{i,j}$ and B_j^λ are in the first and second stage, respectively, the two-stage robust-counterpart of the Wilson model for 2RPFS can be formulated as follows:

$$\min\ y \tag{3.38}$$

$$\text{st } B_n^\lambda + \sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta_{2,i}^\lambda\right) Z_{i,n} \leq y, \qquad \lambda \in \Lambda, \tag{3.39}$$

$$\textstyle\sum_{i=1}^{n}\sum_{\ell=1}^{j} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^\lambda\right) Z_{i,\ell} \leq B_j^\lambda, \qquad j = 1,\ldots,n, \lambda \in \Lambda, \tag{3.40}$$

$$B_j^\lambda + \textstyle\sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta_{2,i}^\lambda\right) Z_{i,j} \leq B_{j+1}^\lambda, \qquad j = 1,\ldots,n-1, \lambda \in \Lambda, \tag{3.41}$$

$$\textstyle\sum_{i=1}^{n} Z_{i,j} = 1, \qquad j = 1,\ldots,n, \tag{3.42}$$

$$\textstyle\sum_{j=1}^{n} Z_{i,j} = 1, \qquad i = 1,\ldots,n, \tag{3.43}$$

$$Z_{i,j} \in \{0,1\}, \qquad i,j = 1,\ldots,n, \tag{3.44}$$

$$B_j^\lambda \geq 0, \qquad j = 1,\ldots,n, \lambda \in \Lambda, \tag{3.45}$$

$$y \geq 0. \tag{3.46}$$

The objective function (3.38) and constraint (3.39) state that this formulation aims to find a robust schedule that minimizes the *makespan* y of the worst-case scenario, among all possible scenarios. Constraints (3.40) and (3.41) guarantee that the robust schedule is feasible and that start time variables are appropriately calculated for each scenario λ. Constraints (3.42) and (3.43) are as defined in the previous formulation. Constraints (3.44)-(3.46) define the domain of the variables.

Solving the two models above, for all possible combinations of $\lambda \in \Lambda$, is unrealistic. Therefore, in the next section, we will introduce an algorithm capable of obtaining optimal results for 2RPFS with only a subset of these combinations.

3.5 Column-and-Constraint Generation applied to 2RPFS problem

This section presents an exact method for solving 2RPFS under budgeted uncertainty. Our approach is based on Column-and-Constraint Generation, a cutting plane procedure for two-stage RO problems which has been recently used to solve different robust scheduling problems [117, 124]. The method's name originated from how the decomposition operates: besides new constraints, each cutting plane is also associated with a set of new decision variables for the recourse problem [162].

Given one of the robust counterparts presented in Section 3.4, the main idea is to relax it into a master problem (MP) where each robust constraint is written only for a finite subset $\mathcal{U}'$ of the uncertainty set $\mathcal{U}_\Gamma$. Then, given a feasible solution to the MP, the solution is checked for feasibility by solving an adversarial separation subproblem (SP). If the SP solution indicates that one or more robust constraints become infeasible, the uncertainty set $\mathcal{U}'$ is expanded by one or more vectors, and the master problem is augmented, according to the column-and-constraint generation procedure.

For the 2RPFS problem, the adversarial separation problem is represented by the worst-case procedure, which, given the sequence σ returned by the MP solution, returns the highest possible *makespan* under the uncertainty set $\mathcal{U}_\Gamma$. Since the uncertainty set $\mathcal{U}_\Gamma$, defined in Section 3.4, is polyhedral, the number of possible extreme solutions that the procedure can fetch is finite, and the algorithm terminates [162].

3.5.1 C&CG algorithm

In order to explain the C&CG algorithm, we will consider the 2-stage RO formulations defined in Section 3.4.3. Given that uncertainty set $\mathcal{U}_{(\Gamma_1,\Gamma_2)}$ is discrete and finite, obtaining a solution for one of these formulations is equivalent to solving a probably large-scale MILP, enumerating all variables and constraints for each scenario λ in the set Λ. This possibility, as we can expect, is unrealistic. [162] propose an alternative solution approach, generating only a subset of scenarios $\Theta = \{\lambda_1, \ldots, \lambda_v\} \subseteq \Lambda$. With the application of the so-called C&CG procedure, if the problem is formulated in a master-subproblem framework, it can be solved iteratively, with each iteration generating one scenario $\lambda_v \in \Theta$, obtained by solving a subproblem.

With this idea in mind, we define the Master Problem (MP) by choosing an appropriate 2-stage RO formulation, in our case, either Wagner or Wilson robust counterpart models. Let $\mathcal{R}_{Wagner} = \{\mathrm{X}^{(1)}, \ldots, \mathrm{X}^{(v)}, \mathrm{Y}^{(1)}, \ldots, \mathrm{Y}^{(v)}\}$ and $\mathcal{R}_{Wilson} = \{\mathrm{B}^{(1)}, \ldots, \mathrm{B}^{(v)}\}$ be the corresponding recourse decision variables of each model, respectively. The master problem is solved iteratively, with each step generating Wagner constraints (3.30)-(3.32) or Wilson constraints (3.39)-(3.41), as well as recourse variables linked with one or more scenarios $\lambda_v \in \Lambda$, obtained by solving the associated subproblem.

In order to deal with the scenarios defined by Θ, we assume that an oracle can obtain an optimal solution to the worst-case subproblem for a given value of the first-stage decision variables $\mathrm{Z}_{i,j}$. The

Algorithm 2: Column-and-constraint generation algorithm

Set $LB = -\infty, UB = +\infty, v = 1$ and $\Theta = \{\lambda_{(0)} : \delta^{(0)}_{r,i} = 0, \forall r = 1, 2, \forall i = 1, \ldots, n\}$
while (UB − LB)/LB $> \epsilon$ **do**
  if $model = Wagner$ **then** Solve the MP defined in (3.29)-(3.37) with $\Lambda := \Theta$
  if $model = Wilson$ **then** Solve the MP defined in (3.38)-(3.46) with $\Lambda := \Theta$
  Let $(\mathrm{Z}^*_{(v)},\ y^*,\ \mathcal{R}^*_{model})$ be the MP optimal solution
  Update LB := max $\left[\mathrm{LB},\ y^*\right]$
  Call the oracle to solve subproblem (SP) in (3.47) with $Z := \mathrm{Z}^*_{(v)}$
  Let $\mathcal{S}^*_{(v)}$ be the SP optimal solution value with associated scenario $\lambda^*_{(v)}$
  Update UB := min $\left[\mathrm{UB},\ \mathcal{S}^*_{(v)}\right]$
  if (UB − LB)/LB $> \epsilon$ **then**
    Add recourse decision variables $\mathcal{R}^{(v)}_{model}$ for scenario $\lambda^*_{(v)}$ on MP
    if $model = Wagner$ **then** Generate MP constraints (3.30)-(3.32)&(3.36) for $\lambda^*_{(v)}$
    if $model = Wilson$ **then** Generate MP constraints (3.39)-(3.41)&(3.45) for $\lambda^*_{(v)}$
    Update $\Theta := \Theta \cup \{\lambda^*_{(v)} : \delta^{(v)} = \delta^*\}$ and set $(v) := (v + 1)$
  end
end
Return UB, $\mathrm{Z}^*_{(v)}$

subproblem SP is defined as:

$$\text{(SP)}\ \ \mathcal{S}(\sigma) = \max_{\lambda_v \in \mathcal{U}_\Gamma} \varphi(\sigma, \mathbf{P}^{\lambda_v}) \tag{3.47}$$

where job permutation σ is derived using (MP) optimal values of variables $Z_{i,j}$. In our case, the optimal solution for (SP) can be obtained by the worst-case procedure defined in Section 3.5.2.

The C&CG method is presented in Algorithm 2, where LB denotes the lower bound, UB denotes the upper bound, v is the iteration counter, Θ is the set of worst-case scenarios generated by the method, and $\epsilon \in \mathbb{R}^+$ represents the tolerance of optimality.

3.5.2 Worst-case evaluation

We now discuss how to determine the worst-case realization under the budgeted uncertainty set $\mathcal{U}_\Gamma$, for a specific sequence of jobs $\sigma = \{\sigma(j), j = 1, \ldots, n\}$. From equation (3.27), given a protection level $\Gamma = (\Gamma_1, \Gamma_2)$ and a schedule σ, we extend the definition of *worst-case makespan* or *robust cost* $\mathcal{Z}(\sigma, \Gamma)$ as follows:

$$\mathcal{Z}(\sigma, \Gamma) := \max_{\lambda \in \mathcal{U}_\Gamma} \{\varphi(\sigma, \mathbf{P}^{\lambda})\}. \tag{3.48}$$

We assume that parameters Γ_1 and Γ_2, from the budgeted uncertainty set, are non-negative integers. Based on this assumption, statement (3.48) reflects a problem with a convex function being maximized over a polytope defined by uncertainty set $\mathcal{U}_\Gamma$. Thus, in order to obtain the worst-case realization of uncertainty, only specific realizations of $\mathcal{U}_\Gamma$ are needed, namely the extreme points of the polytope. For each machine M_r and job J_i, the set of extreme scenarios are characterized either by the values $\overline{p}_{r,i}$ or $\overline{p}_{r,i} + \widehat{p}_{r,i}$ (see proof in Appendix C), i.e., any worst-case realization will use as

much budget of uncertainty as possible. Therefore, for the optimal solution of (3.48), with worst-case scenario λ^*, $\sum_{i=1}^{n} \frac{|p_{r,i}^{\lambda^*} - \overline{p}_{r,i}|}{\widehat{p}_{r,i}} = \Gamma_r, \forall r \in \{1,2\}$.

We developed a worst-case solution method based on dynamic programming. The complexity of this algorithm is $\mathcal{O}(n^2)$. Given $1 \leq r \leq 2,\ 1 \leq k \leq n$, and $0 \leq \gamma \leq n$, let us define a value function $\alpha(r,k,\gamma)$ as the optimal value of the restricted separation problem for machine M_r and job positions $\{1,\ldots,k\}$, when at most γ jobs are using their maximum processing time on machine M_r. The optimal value of the problem is then defined by $\mathcal{Z}(\sigma,\Gamma) = \alpha(2,n,\Gamma_2)$.

The value-function is defined by the recursion:

$$\begin{aligned}\alpha(1,k,\gamma) = \max\Big[&\overline{p}_{1,\sigma(k)} + \alpha(1,\ k-1,\ \gamma),\ \overline{p}_{1,\sigma(k)} + \widehat{p}_{1,\sigma(k)} + \alpha(1,\ k-1,\ \gamma-1)\Big],\\ &for\ 1 \leq k \leq n,\ 0 \leq \gamma \leq \Gamma_1, \qquad (3.49)\\ \alpha(2,k,\gamma) = \max\Big[&\overline{p}_{2,\sigma(k)} + \max[\alpha(2,\ k-1,\ \gamma),\ \alpha(1,\ k,\ \Gamma_1)]\ ,\\ &\overline{p}_{2,\sigma(k)} + \widehat{p}_{2,\sigma(k)} + \max[\alpha(2,\ k-1,\ \gamma-1),\ \alpha(1,\ k,\ \Gamma_1)]\Big],\\ &for\ 1 \leq k \leq n,\ 0 \leq \gamma \leq \Gamma_2, \qquad (3.50)\end{aligned}$$

and the following initialization values:

$$\alpha(r,k,\gamma) = -\infty\ if\ \gamma < 0,\quad \alpha(r,k,\gamma) = 0\ if\ k = 0 \text{ and } \gamma \geq 0,$$
$$\alpha(2,k,0) = \overline{p}_{2,\sigma(k)} + \max[\alpha(2,\ k-1,\ 0),\ \alpha(1,\ k,\ \Gamma_1)],\ for\ 1 \leq k \leq n.$$

If $r = 1$, the first maximizer argument accounts for the case when there is no delay of execution regarding job $\sigma(k)$ on the first machine, while the second expression handles the case where a delay occurs. Similarly, for $r = 2$, we take the maximum of two cases: with or without delay when executing job $\sigma(k)$ on the second machine. However, the job start time has to be computed as the maximum between the previous job's $\sigma(k-1)$ worst-case completion time on the same machine M_2, and the worst-case completion time of the same job $\sigma(k)$ on the previous machine M_1, taking into account its budget of uncertainty Γ_1.

3.6 2RPFS Experimental results

We conducted extensive experiments on randomly generated datasets to assess the performance of the proposed solution method and additional aspects, including solution robustness, the trade-off between robustness and optimality, and the impact of data uncertainty on the obtained schedules. The analyses employed in this section follow the same lines as recent works on RO under budget uncertainty [93, 36, 124]. Sub-section 3.6.1 presents the testbed and environment setup, while 3.6.2 examines the robust method performance regarding execution time and the number of optimal solutions. Finally, based on Monte-Carlo simulation, we close with a case study that analyses the expected behavior of robust, stochastic, and deterministic solutions, to verify a possible increase in the expected solution cost in the long run.

3.6.1 Test instances and computational environment

Our experiments were based on a set of random instances generated by [159]. In his work, six groups of instances were created, each one with a different number of jobs $n = \{10, 20, 50, 100, 150, 200\}$. The expected processing time $\overline{p}_{r,i}$ $(r = 1, 2;\ i = 1, \ldots, n)$ is an integer drawn from the uniform distribution [10, 50] and the largest processing time deviation was set as a ratio of the expected processing time (i.e., $\widehat{p}_{r,i} = \alpha \overline{p}_{r,i}$), where $\alpha = \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Ten instances were generated for each combination of n and α for a total of 300 test instances. All test instances are available at https://github.com/levorato/2RPFS_Cmax_Budget.

The C&CG algorithm was coded in Julia 1.4.0, and IBM CPLEX 12.9.0 (with default parameters) was used to solve 2RPFS MILP models. All experiments were performed on a workstation with an Intel Xeon® CPU E5640 @2.67GHz with 32 GB RAM, under Ubuntu 18.04 LTS. Time limit was set to 2 hours to solve each instance and the ϵ convergence parameter for C&CG was set to 10^{-8}.

With a particular interest in examining the impact of budget parameters on the performance of the proposed robust scheduling algorithms, when solving each instance, we tested the 2RPFS models by varying Γ_1 and Γ_2 according to five ratios (20%, 40%, 60%, 80%, and 100%) of the number of operations subject to processing time deviation on machines M_1 and M_2, respectively.

3.6.2 Comparative performance of the algorithms

This section examines the performance and effectiveness of the C&CG algorithm when using either Wagner or Wilson 2RPFS models. The comparison is based on the computational efficiency in terms of CPU time and the percentage of instances solved to optimality (i.e., zero solution gap).

We present in Table 3.2 overall results, comparing the performance of the algorithms. Wagner-model C&CG is the one that solves the majority of the instances to optimality with the best execution time. The *% Best Performance* measurement indicates that, from the total number of instances solved to optimality, the Wagner model solved 86% of these instances faster, using less CPU time, followed by Wilson, which solved 14%. Measurements *% Solved 150* and *% Solved 200* indicate that the Wilson-based algorithm could not obtain optimal solutions for most of the 150 and 200-job instances within the time limit. The other presented measurements (*%Solved, Avg % Gap*, and *Median time*) also favor the Wagner model. In this analysis, we present medians to mitigate the effect of instances not solved within the time limit.

Table 3.3 presents, for every instance size, the average performance of the C&CG algorithm with each robust-counterpart model, including average run time values. When using average, the results of all instances (even outliers) are taken into account. Standard deviation is also included as a secondary measure. Additionally, the average number of iterations and the standard deviation are listed. These results evidence that, as instance size grows, the models become harder to solve (especially the Wilson model), as seen on the smaller percentage of instances solved to optimality

Model	%Best Performance	% Solved	% Solved 10-20	% Solved 50	% Solved 100	% Solved 150	% Solved 200	Avg. % gap	Median time	Median Iterations
Wagner	86%	55%	100%	97%	90%	68%	67%	1.11%	14.05	9.0
Wilson	14%	45%	100%	86%	58%	46%	37%	1.21%	148.62	7.0

Table 3.2 – Wagner *vs.* Wilson Robust PFSP C&CG performance comparison, given all instances. % Best Performance is the percentage of instances solved to optimality where the model achieved shorter execution time; % Solved contains the percentage of instances solved to optimality within the time limit; % Solved $< n >$ represents the percentage of solved instances of size n; Avg. % gap is the average percentage gap of solutions from instances not solved to optimality; Median time is the median execution time, in seconds; Median iterations is the median of the number of iterations performed.

	10		20		50		100		150		200	
	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson
% Best Performance	61%	39%	81%	19%	95%	5%	95%	5%	93%	7%	95%	5%
% Solved	100%	100%	100%	100%	97%	86%	90%	58%	68%	46%	67%	37%
Avg. % gap					0.40%	0.86%	1.09%	1.68%	1.21%	1.62%	1.09%	0.61%
Avg. time opt. (s)	0	0	2	7	117	310	258	526	338	878	441	2,362
Std. dev. of time opt. (s)	1	1	7	54	616	888	684	1,177	877	1,233	752	1,888
Avg. Iterations	4	4	7	7	17	20	31	21	28	14	30	11
Std. dev. of Iterations	2	2	8	8	24	23	32	15	28	8	44	4

Table 3.3 – Wagner *vs.* Wilson Robust PFSP C&CG performance comparison, for each instance size n. % Best Performance is the percentage of instances solved to optimality where the model achieved shorter execution time; % Solved contains the percentage of instances solved to optimality within the time limit; Avg. % gap is the average percentage gap of solutions from instances not solved to optimality; Avg. time opt. and Std. dev. of time opt. are the mean and standard deviation in solution time, respectively, regarding instances solved to optimality; Avg. iterations and Std. dev. of iterations are the mean and standard deviation of the number of iterations performed.

and increased average execution time.

3.6.3 Case study on two representative instances

In this subsection, we assess the quality and level of robustness of scheduling solutions for two large problem instances, the first one with small uncertainty ($\alpha = 20\%$) and the second with high uncertainty ($\alpha = 50\%$). The following solution methods were used:

— **Det(P=)**: deterministic PFSP solution with $\mathbf{P} = \{\overline{p}_{r,i}\}, \forall r \in \mathbb{M}, i \in \mathbb{J}$;

— **2RPFS(**Γ_1, Γ_2**)**: Wagner-based 2RPFS model, solved with the C&CG framework. The Γ parameters are used to control the level of the conservativeness of the robust model, and both vary in the set $\{20\%, 40\%, 60\%, 80\%, 100\%\}$ as a fraction of the number of jobs n. The robust model with $\Gamma_1 = \Gamma_2 = 0$ is equivalent to **Det(P=)**, while the one with $\Gamma_1 = \Gamma_2 = n$ is the deterministic model that is entirely risk-averse and overestimates all parameters. The other values of Γ_1 and Γ_2 model intermediate risk aversions;

— **SimGRASP**: stochastic PFSP simheuristic solution from [40]. SimGRASP is a modified GRASP metaheuristic that incorporates Monte Carlo Simulation to solve the PFSP with random processing times. The objective is to find a schedule that minimizes the expected makespan. Given its stochastic nature, we obtained 25 independent runs for each instance file (and respective α parameter). For result comparison, when calculating the robust cost of each (Γ_1, Γ_2) combination, we stored, for each instance, the smallest and largest robust costs found within these 25 simheuristic executions. We call them **SimGRASP-Min(25)** and **SimGRASP-Max(25)**.

We assessed the robustness of each solution method by calculating the *robust cost* at different protection levels $(\Gamma_1\%, \Gamma_2\%)$, using the dynamic programming algorithm defined in Section 3.5.2. Figure 3.3 depicts the *robust cost* $\mathcal{Z}(\sigma)$ of each solution σ under different protection levels $(\Gamma_1\%, \Gamma_2\%)$. For clarity of the graphs, the robust costs for some protection levels were omitted.

Observe that, as the protection level $(\Gamma_1\%, \Gamma_2\%)$ increases, so does the robust cost, i.e., *makespan* of the worst-case scenario defined by the protection level. In other words, higher values of $\Gamma_1\%$ and $\Gamma_2\%$ are equivalent to a greater quantity of operations with deviated processing times, which directly impacts the *makespan*. In the examples from Figure 3.3, the extreme cases occur whenever $\Gamma_2\% = 100$, yielding the highest robust costs.

From the viewpoint of the decision-maker who needs to hedge against worst-case costs, it would be preferable to obtain a solution method that performs well under different protection levels. With this in mind, in the two graphs presented, we identify which scheduling method (and respective solution) presents the best (smallest) robust cost, considering all $(\Gamma_1\%, \Gamma_2\%)$ values. Regarding the first graph (small uncertainty instance), note that both **2RPFS(80,40)** and **2RPFS(60,40)** offer improved protection against worst-case scenarios, regardless of $(\Gamma_1\%, \Gamma_2\%)$ values used for worst-case evaluation. We also highlight the disappointing worst-case performance of both the nominal solution **Det(=)** and the stochastic method. The vast distance between the robust costs of the stochastic method, i.e., **SimGRASP-Min(25)** and **SimGRASP-Max(25)**, reveals a significant exposure to the realization of worst-case scenarios.

In its turn, the "large uncertainty range" instance ($\alpha = 50\%$) presents increased robust cost differences between distinct protection levels. For this instance, the variation of Γ_1 and Γ_2, i.e., the number of operations whose processing times can deviate on each machine, has even more impact on the worst-case makespan. In Figure 3.3(b), we can observe that either **2RPFS(40,40)** or **2RPFS(60,60)** offer the best protection against worst-case scenarios, depending on the combination of $(\Gamma_1\%, \Gamma_2\%)$ values. Once again, the solutions **Det(=)** and **SimGRASP-Max(25)** present high robust costs. In particular, for $(\Gamma_1\%, \Gamma_2\%) = (60, 60)$, the solution provided by **2RPFS(60,60)** is 8% cheaper than **Det(=)** and **SimGRASP-Max(25)**.

In summary, the choice of a robust solution depends on the instance and the desired protection level. The examples above illustrate how 2RPFS can provide a pool of robust schedules, depending on the value of (Γ_1, Γ_2). With these options, the decision-makers can choose one of the schedules

based on their risk preferences. Also, remark that, if the stochastic method is chosen, depending on the solution returned by the algorithm, the worst-case performance may be weak, as can be seen on the robust costs achieved by **SimGRASP-Max(25)**. Indeed, neither **SimGRASP** nor the deterministic models have the objective of minimizing the *worst-case makespan*.

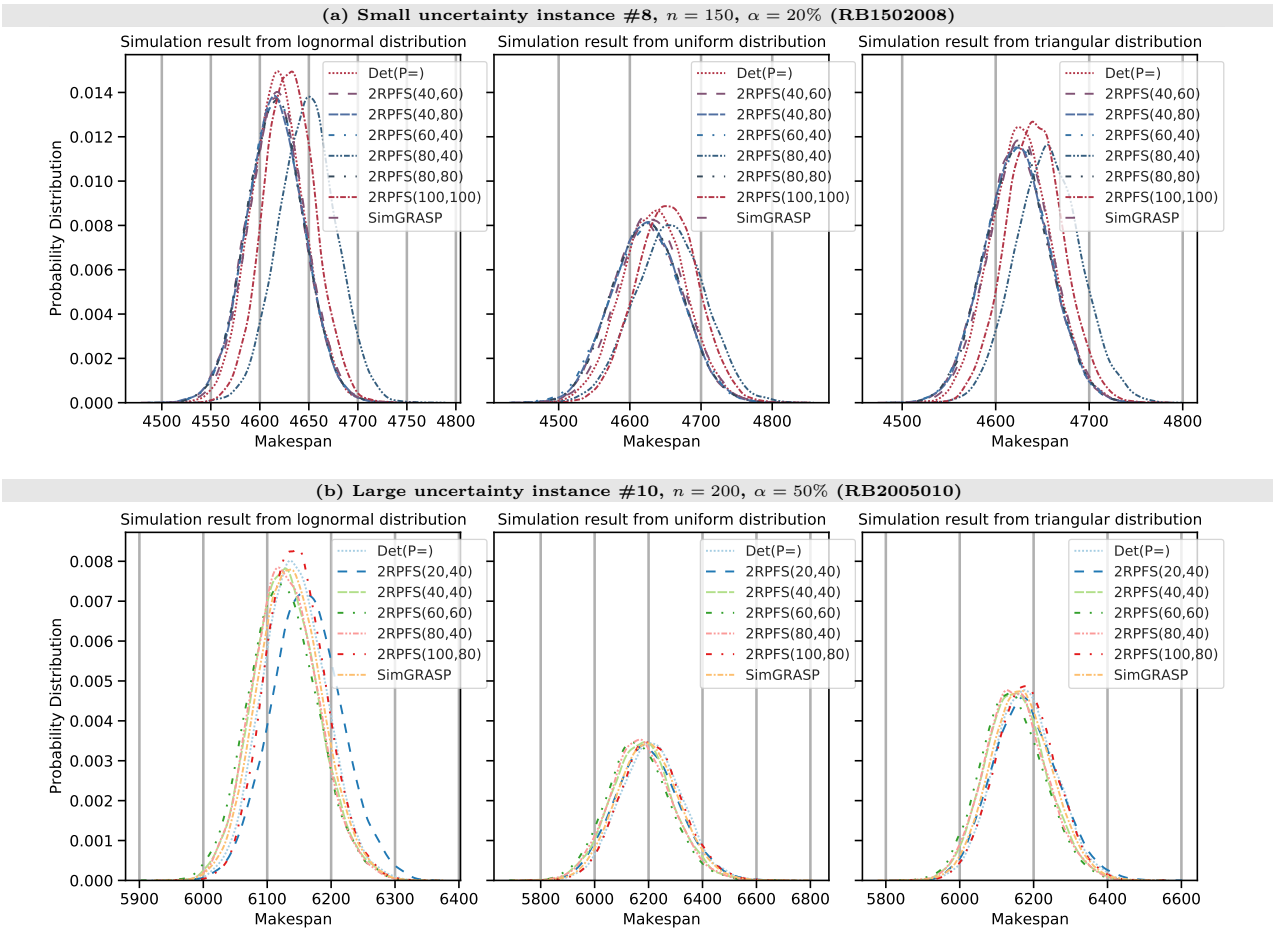

Figure 3.2 – Probability distributions of *makespan* value for 2RPFS and SimGRASP solutions, according to simulation results from lognormal, triangular, and uniform distributions for uncertain job processing times.

As a complementary analysis, we evaluate the expected behavior of obtained problem solutions. The *makespan* distribution of the obtained robust schedules was simulated by subjecting the processing time matrix to random perturbations. In particular, in each Monte Carlo simulation run, the (actual) processing time $\widetilde{p}_{r,i}, \forall r \in \mathbb{M}, i \in \mathbb{J}$, was independently drawn from a predefined probability distribution, yielding a random processing time matrix $\widetilde{P}$. For this purpose, we used lognormal, symmetric triangular, and uniform distributions in $[\overline{p} - \widehat{p}, \overline{p} + \widehat{p}]$ to generate random processing times. We generated $10,000$ processing time matrices $\widetilde{P}$. Then, for each **2RPFS** solution $\sigma^{(\Gamma_1,\Gamma_2)}$, obtained with a specific protection level (Γ_1, Γ_2), we processed the set of all corresponding *makespan* values

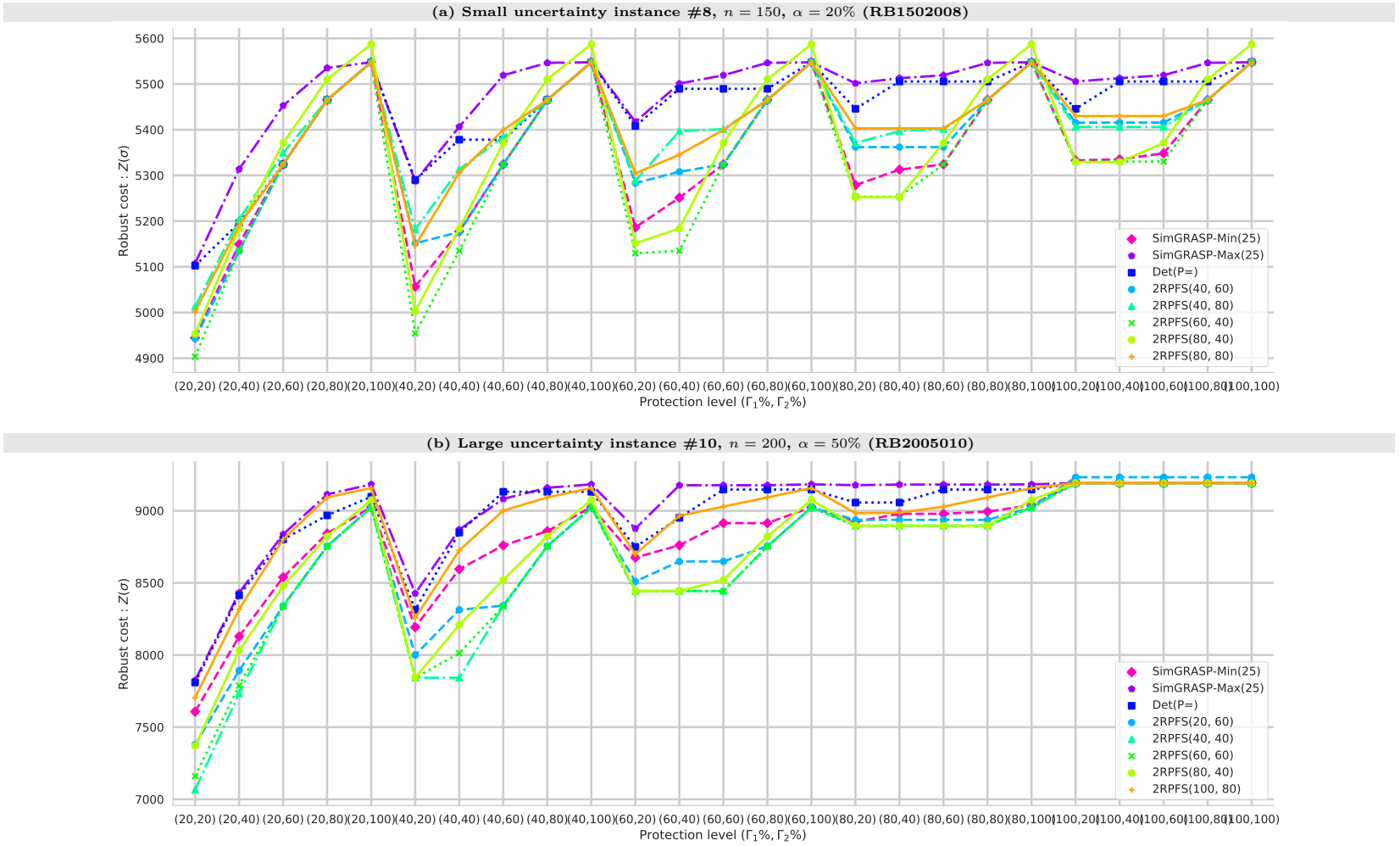

Figure 3.3 – Robust cost of deterministic, 2RPFS and SimGRASP solutions versus protection level (Γ_1%, Γ_2%). All presented 2RPFS solutions are optimal.

$\varphi(\sigma^{(\Gamma_1,\Gamma_2)}, \widetilde{P})$ obtained through simulation on $\widetilde{P}$. The same was made for the solutions returned by **Det(P=)** and **SimGRASP-Min(25)**.

We first focus on simulation results presented in Figure 3.2(a). Regarding the small uncertainty instance, the expected makespan performance of **2RPFS(40,60)**, **2RPFS(40,80)**, **2RPFS(60,40)** and **2RPFS(80,80)**, are equivalent to **SimGRASP**. However, depending on the budget parameters, if we return to worst-case evaluation, as seen in Figure 3.3(a), the protection against worst-case scenarios varies considerably. The best performing solutions, from smallest to largest robust cost, are: **2RPFS(60,40)**, **2RPFS(80,40)**, **SimGRASP-Min(25)** and **2RPFS(40,60)**. When analyzing the large uncertainty instance in Figure 3.2(b), the following robust solutions present expected makespan performance quite similar to **SimGRASP**: **2RPFS(40,40)**, **2RPFS(60,60)** and **2RPFS(80,40)**. However, according to the worst-case evaluation, only the first two provide better protection against worst-case costs.

Finally, Table 3.4 presents some statistics related to the simulation of processing times of the large uncertainty instance. Given $10,000$ processing time matrices $\widetilde{P}$ obtained after simulation runs, let $\varphi(\sigma)$ be the random cost (*makespan*) of scheduling σ, which depends on the realization of P. $E(\varphi(\sigma))$ and $SD(\varphi(\sigma))$ are empirical estimations of expectation and standard deviation of $\varphi(\sigma)$, respectively. Also, $\varphi^{0.95}(\sigma)$ and $\varphi^{0.99}(\sigma)$ are the 0.95 and 0.99 quantiles of $\varphi(\sigma)$, respectively, and $\varphi^{max}(\sigma)$ is the maximum observed $\varphi(\sigma)$ in the simulation.

Observe that **2RPFS(60,60)** has the least $E(\varphi(\sigma))$ in lognormal distribution, while **2RPFS(40,20)** presents the smallest expected makespan in symmetric triangular and uniform distributions. When analyzing the largest observed makespan, **2RPFS(60,60)**, **2RPFS(80,80)**, and **2RPFS(80,80)** have the **lowest** $\varphi^{max}(\sigma)$ for lognormal, triangular and uniform distributions, respectively. The best solutions for **Det(P=)** and **SimGRASP** did not provide minimum values for any measure of the simulated distributions. Also, by analyzing the smallest maximum makespan obtained in uniform distribution simulations, the value $\varphi^{max}(\sigma)$ observed for scheduling **2RPFS(80,80)** is 2% cheaper than **SimGRASP**, and, at the same time, its expected makespan is 0.5% less than the stochastic schedule. Based on these observations, the hedge provided by the obtained robust solutions does not cause a significant increase in the expected solution cost when compared to stochastic and deterministic solutions.

3.6.4 Evaluating price of robustness and hedge value

Given a protection level Γ, besides robust cost $\mathcal{Z}$, two other measures can be used to evaluate performance: price of robustness η and hedge value H.

$$\eta(\Gamma) = \varphi(\sigma^*_\Gamma, \overline{P}) - \varphi(\bar{\sigma}^*, \overline{P}), \tag{3.51}$$

$$H(\Gamma) = \mathcal{Z}(\bar{\sigma}^*, \Gamma) - \mathcal{Z}(\sigma^*_\Gamma, \Gamma), \tag{3.52}$$

where $\varphi(.)$ is the *makespan* function, σ^*_Γ is the optimal solution of **2RPFS(Γ_1, Γ_2)** and $\bar{\sigma}^*$ is the optimal solution of **Det(P=)**.

Large uncertainty instance #10, $n = 200$, $\alpha = 50\%$ (RB2005010)

	Lognormal Distribution					Symmetric Triangular Distribution					Uniform Distribution				
Method	E($\varphi(\sigma)$)	SD($\varphi(\sigma)$)	$\varphi^{0.95}(\sigma)$	$\varphi^{0.99}(\sigma)$	$\varphi^{max}(\sigma)$	E($\varphi(\sigma)$)	SD($\varphi(\sigma)$)	$\varphi^{0.95}(\sigma)$	$\varphi^{0.99}(\sigma)$	$\varphi^{max}(\sigma)$	E($\varphi(\sigma)$)	SD($\varphi(\sigma)$)	$\varphi^{0.95}(\sigma)$	$\varphi^{0.99}(\sigma)$	$\varphi^{max}(\sigma)$
2RPFS(20,20)	6,125.6	52.4	6,214.5	6,254.6	6,320.1	6,143.6	84.6	6,288.4	6,352.5	6,467.6	6,162.2	116.2	6,360.4	6,446.8	6,627.1
2RPFS(20,40)	6,161.4	54.3	6,252.7	6,291.4	6,348.8	6,177.5	88.7	6,328.5	6,397.0	6,559.7	6,195.0	119.7	6,400.8	6,483.0	6,688.0
2RPFS(20,60)	6,151.5	53.8	6,241.8	6,280.4	6,342.5	6,165.5	87.0	6,312.7	6,379.8	6,499.7	6,182.0	118.7	6,383.1	6,472.4	6,677.8
2RPFS(20,80)	6,164.1	54.5	6,256.3	6,294.4	6,384.2	6,176.8	88.5	6,328.6	6,394.3	6,561.2	6,191.3	119.7	6,394.8	6,491.6	6,703.6
2RPFS(20,100)	6,211.5	54.9	6,302.3	6,340.8	6,444.0	6,228.4	93.9	6,385.6	6,450.2	6,571.4	6,247.2	126.0	6,455.8	6,544.9	6,709.3
2RPFS(40,20)	6,125.6	52.5	6,214.6	6,254.3	6,317.2	6,143.0	84.8	6,288.5	6,353.4	6,476.9	6,160.9	116.5	6,359.3	6,445.3	6,639.6
2RPFS(40,40)	6,129.9	50.7	6,215.7	6,257.6	6,321.3	6,153.9	83.2	6,294.9	6,358.3	6,466.5	6,177.3	114.8	6,371.5	6,462.6	6,667.6
2RPFS(40,60)	6,142.7	53.4	6,232.8	6,273.3	6,347.5	6,157.7	86.6	6,305.5	6,372.1	6,493.2	6,174.6	118.0	6,374.2	6,468.2	6,694.6
2RPFS(40,80)	6,140.7	53.2	6,229.8	6,268.7	6,326.6	6,159.3	86.3	6,307.6	6,372.7	6,508.4	6,178.8	117.1	6,376.1	6,460.3	6,636.4
2RPFS(40,100)	6,234.3	54.4	6,324.3	6,358.5	6,431.3	6,253.3	92.9	6,409.4	6,472.9	6,631.6	6,273.7	128.1	6,485.6	6,573.4	6,740.0
2RPFS(60,20)	6,127.5	51.3	6,214.7	6,254.9	6,336.5	6,147.4	83.8	6,291.9	6,354.5	6,529.1	6,167.7	114.9	6,360.5	6,447.1	6,681.4
2RPFS(60,40)	6,129.1	50.1	6,215.2	6,252.7	6,315.6	6,151.4	82.8	6,291.6	6,357.5	6,491.5	6,172.1	114.0	6,364.7	6,451.6	6,627.3
2RPFS(60,60)	6,125.4	52.5	6,214.4	6,254.2	6,313.5	6,143.5	84.5	6,287.4	6,350.5	6,479.8	6,161.3	115.2	6,353.3	6,446.8	6,643.0
2RPFS(60,80)	6,164.2	54.3	6,255.4	6,295.1	6,362.1	6,176.7	88.8	6,330.5	6,394.2	6,518.9	6,191.9	119.9	6,396.2	6,486.2	6,711.7
2RPFS(60,100)	6,138.8	52.8	6,228.6	6,266.2	6,335.9	6,163.4	85.1	6,307.0	6,372.0	6,506.3	6,189.2	117.6	6,386.3	6,473.2	6,628.6
2RPFS(80,20)	6,269.6	54.6	6,360.7	6,400.4	6,483.5	6,289.8	95.0	6,446.2	6,512.8	6,711.9	6,309.8	129.0	6,524.2	6,613.2	6,884.9
2RPFS(80,40)	6,129.4	50.0	6,214.0	6,253.1	6,318.0	6,151.9	82.9	6,294.5	6,354.0	6,484.0	6,173.1	114.3	6,365.7	6,456.4	6,687.7
2RPFS(80,60)	6,128.7	50.3	6,215.1	6,252.1	6,334.1	6,152.2	82.8	6,291.2	6,355.0	6,478.6	6,175.0	114.3	6,367.7	6,454.8	6,660.6
2RPFS(80,80)	6,126.8	51.7	6,214.3	6,254.8	6,320.3	6,147.4	84.2	6,291.7	6,353.3	6,464.4	6,168.1	114.9	6,361.4	6,451.5	6,623.2
2RPFS(80,100)	6,148.4	53.9	6,238.9	6,276.7	6,345.9	6,167.0	86.7	6,315.8	6,379.3	6,497.7	6,187.8	118.0	6,387.2	6,483.1	6,695.6
2RPFS(100,20)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
2RPFS(100,40)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
2RPFS(100,60)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
2RPFS(100,80)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
2RPFS(100,100)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
Det(P =)	6,142.8	49.0	6,223.8	6,264.4	6,325.0	6,178.0	82.4	6,316.9	6,375.9	6,523.4	6,211.0	114.4	6,399.8	6,485.5	6,626.0
SimGRASP	6,136.2	50.6	6,221.8	6,259.3	6,366.0	6,167.4	83.8	6,309.0	6,370.4	6,544.3	6,198.1	115.7	6,391.8	6,477.7	6,783.3

Table 3.4 – Simulation summary for 2RPFS, Det(P=), and SimGRASP solution methods from lognormal, triangular, and uniform distributions of processing times. Minimum values for each column are highlighted.

The first measure, $\eta(\Gamma)$, is defined as the price paid by the decision-maker for employing the robust sequence σ^*_Γ in place of the optimal nominal sequence $\bar{\sigma}^*$ in the scenario of nominal processing times (when $P = \overline{P}$, i.e., no processing time deviations). $H(\Gamma)$ represents the value gained from adopting the robust sequence σ^*_Γ, instead of the optimal nominal sequence $\bar{\sigma}^*$ in the occurrence of the worst-case scenario associated with protection level $\Gamma = (\Gamma_1, \Gamma_2)$. In other words, $\eta(\Gamma)$ can be seen as the *trade-off between robustness and optimality*, and $H(\Gamma)$ represents the *regret of employing sequence $\bar{\sigma}^*$ in the worst-case scenario*.

Table 3.5 displays the relative price of robustness $\eta(\Gamma)\% = \frac{\varphi(\sigma^*_\Gamma,\overline{P}) - \varphi(\bar{\sigma}^*,\overline{P})}{\varphi(\bar{\sigma}^*,\overline{P})}$ and hedge value $H(\Gamma)\% = \frac{\mathcal{Z}(\bar{\sigma}^*,\Gamma) - \mathcal{Z}(\sigma^*_\Gamma,\Gamma)}{\mathcal{Z}(\sigma^*_\Gamma,\Gamma)}$ for various protection levels, based on instance #8 with $n = 150$, with different degrees of processing time uncertainty α. Observe that, given a protection level Γ, as α grows, so does the regret $H(\Gamma)\%$ of employing the optimal nominal sequence in the occurrence of the worst-case scenario defined by Γ. The only exception is for extreme values of Γ_1 and Γ_2, where $H(\Gamma)\% = 0$. Regarding the relative price of robustness, for several protection levels Γ, the relative robustness price $\eta(\Gamma)\%$ is zero, i.e., in the absence of processing time deviations, most robust schedules present the same makespan as the optimal nominal solution. Among these schedules, the best ones, which maximize hedge value $H(\Gamma)\%$, are **2RPFS(60,20)** for $\alpha = 10\%$, **2RPFS(60,40)** for $\alpha \in \{20\%, 30\%, 40\%\}$, and **2RPFS(40,40)** for $\alpha = 50\%$.

		$\Gamma_1 = 20$					$\Gamma_1 = 40$					$\Gamma_1 = 60$					$\Gamma_1 = 80$					$\Gamma_1 = 100$				
		Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100
α = 10%	η %	0.0%	0.0%	0.0%	2.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	H %	1.7%	0.2%	0.0%	0.0%	0.0%	3.7%	2.1%	0.2%	0.0%	0.0%	4.4%	3.3%	1.3%	0.0%	0.0%	3.4%	3.4%	1.5%	0.1%	0.0%	2.6%	3.3%	1.5%	0.1%	0.0%
α = 20%	η %	0.0%	0.0%	0.0%	0.3%	5.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.9%	0.0%	0.0%	0.0%	0.4%	0.7%	0.7%	0.0%	0.0%	0.6%	1.5%	1.0%	0.0%	0.0%	0.0%
	H %	4.1%	1.2%	0.0%	0.0%	0.0%	6.8%	4.7%	1.0%	0.0%	0.0%	5.4%	6.9%	3.1%	0.4%	0.0%	3.7%	4.8%	3.4%	0.7%	0.0%	2.2%	3.3%	3.3%	0.7%	0.0%
α = 30%	η %	0.0%	0.0%	0.0%	0.0%	3.7%	0.0%	0.0%	0.0%	0.0%	1.3%	0.0%	0.0%	0.0%	0.0%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%
	H %	6.3%	2.1%	0.0%	0.0%	0.0%	8.3%	7.1%	2.1%	0.0%	0.0%	6.4%	8.6%	4.7%	0.9%	0.0%	4.1%	5.4%	5.1%	1.3%	0.0%	2.2%	3.3%	3.3%	1.3%	0.0%
α = 40%	η %	0.0%	0.0%	0.0%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.1%	0.0%	0.0%	0.0%
	H %	8.4%	3.3%	0.6%	0.0%	0.0%	9.7%	9.3%	3.5%	0.0%	0.0%	7.2%	9.9%	6.1%	1.4%	0.0%	4.4%	5.9%	5.9%	1.8%	0.0%	2.2%	3.3%	3.3%	1.8%	0.0%
α = 50%	η %	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.2%	0.1%	0.3%	0.0%	0.0%
	H %	10.4%	4.6%	1.4%	0.0%	0.0%	10.9%	11.3%	4.7%	0.0%	0.0%	7.9%	11.1%	7.4%	1.7%	0.0%	4.7%	6.4%	6.4%	2.3%	0.0%	2.2%	3.3%	3.3%	2.3%	0.0%

Table 3.5 – Relative robustness price $\eta(\Gamma)\%$ and hedge value $H(\Gamma)\%$ for instance #8, $n = 150$, for different degrees of uncertainty $\alpha \in \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Best values are highlighted.

			$\Gamma_1 = 20$					$\Gamma_1 = 40$					$\Gamma_1 = 60$					$\Gamma_1 = 80$					$\Gamma_1 = 100$				
			Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100
α = 10%	logno..	ω	51%	52%	52%	0%	36%	52%	52%	37%	51%	36%	52%	52%	52%	51%	36%	0%	51%	51%	52%	36%	38%	52%	52%	52%	36%
		Δϕ	0.0%	0.0%	0.0%	2.2%	0.1%	0.0%	0.0%	0.1%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%	0.7%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%
	trian..	ω	52%	52%	52%	0%	38%	52%	52%	38%	52%	38%	51%	51%	51%	51%	38%	0%	51%	52%	51%	38%	39%	52%	52%	51%	38%
		Δϕ	0.0%	0.0%	0.0%	2.2%	0.1%	0.0%	0.0%	0.1%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%	0.7%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%
	unifor..	ω	55%	54%	54%	0%	34%	55%	54%	37%	53%	34%	54%	54%	54%	51%	34%	2%	54%	54%	53%	34%	45%	55%	54%	54%	34%
		Δϕ	0.0%	0.0%	0.0%	2.2%	0.1%	0.0%	0.0%	0.1%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%	0.7%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%
α = 20%	logno..	ω	56%	56%	51%	1%	29%	7%	55%	56%	55%	29%	55%	56%	56%	55%	29%	48%	56%	56%	56%	29%	48%	42%	56%	56%	29%
		Δϕ	-0.1%	-0.1%	0.0%	1.7%	0.2%	0.4%	-0.1%	-0.1%	-0.1%	0.2%	-0.1%	-0.1%	-0.1%	-0.1%	0.2%	0.0%	-0.1%	-0.1%	-0.1%	0.2%	0.0%	0.0%	-0.1%	-0.1%	0.2%
	trian..	ω	57%	57%	50%	2%	30%	16%	58%	58%	58%	30%	57%	57%	58%	57%	30%	52%	57%	57%	57%	30%	52%	49%	57%	58%	30%
		Δϕ	-0.1%	-0.1%	0.0%	1.6%	0.2%	0.4%	-0.1%	-0.1%	-0.1%	0.2%	-0.1%	-0.1%	-0.1%	-0.1%	0.2%	-0.1%	-0.1%	-0.1%	-0.1%	0.2%	-0.1%	0.0%	-0.1%	-0.1%	0.2%
	unifor..	ω	61%	61%	51%	6%	29%	31%	61%	61%	61%	29%	61%	60%	61%	61%	29%	57%	61%	61%	58%	29%	57%	56%	60%	62%	29%
		Δϕ	-0.2%	-0.2%	0.0%	1.5%	0.3%	0.3%	-0.2%	-0.2%	-0.2%	0.3%	-0.2%	-0.2%	-0.2%	-0.2%	0.3%	-0.2%	-0.2%	-0.2%	-0.1%	0.3%	-0.2%	-0.1%	-0.2%	-0.2%	0.3%
α = 30%	logno..	ω	58%	58%	58%	59%	0%	15%	58%	59%	58%	5%	51%	52%	59%	58%	24%	52%	52%	59%	57%	58%	56%	52%	43%	58%	58%
		Δϕ	-0.1%	-0.1%	-0.1%	-0.1%	3.6%	0.3%	-0.1%	-0.1%	-0.1%	1.2%	-0.1%	-0.1%	-0.1%	-0.1%	0.4%	-0.1%	-0.1%	-0.1%	-0.1%	-0.1%	-0.1%	-0.1%	0.0%	-0.1%	-0.1%
	trian..	ω	61%	61%	62%	61%	0%	36%	62%	62%	61%	11%	58%	58%	62%	61%	29%	58%	58%	62%	59%	57%	58%	58%	55%	62%	57%
		Δϕ	-0.2%	-0.2%	-0.2%	-0.2%	3.5%	0.2%	-0.2%	-0.2%	-0.2%	1.1%	-0.2%	-0.2%	-0.3%	-0.2%	0.4%	-0.2%	-0.2%	-0.2%	-0.2%	-0.1%	-0.2%	-0.2%	-0.1%	-0.2%	-0.1%
	unifor..	ω	65%	63%	62%	62%	1%	47%	64%	64%	63%	17%	62%	63%	65%	63%	31%	62%	63%	65%	59%	56%	61%	62%	61%	63%	56%
		Δϕ	-0.5%	-0.3%	-0.3%	-0.3%	3.3%	-0.1%	-0.4%	-0.4%	-0.3%	1.1%	-0.4%	-0.4%	-0.5%	-0.3%	0.4%	-0.4%	-0.4%	-0.5%	-0.2%	-0.1%	-0.4%	-0.4%	-0.4%	-0.4%	-0.1%
α = 40%	logno..	ω	61%	59%	61%	24%	32%	57%	59%	60%	60%	20%	55%	58%	61%	61%	60%	53%	48%	59%	60%	52%	43%	44%	57%	60%	45%
		Δϕ	-0.2%	-0.2%	-0.2%	0.4%	0.2%	-0.1%	-0.2%	-0.2%	-0.1%	0.6%	-0.1%	-0.1%	-0.2%	-0.2%	-0.1%	-0.1%	-0.1%	-0.2%	-0.2%	-0.1%	0.0%	0.0%	-0.1%	-0.2%	0.1%
	trian..	ω	64%	62%	64%	32%	35%	63%	63%	62%	62%	27%	62%	63%	64%	64%	62%	61%	60%	64%	63%	55%	50%	56%	63%	65%	47%
		Δϕ	-0.4%	-0.3%	-0.4%	0.4%	0.3%	-0.4%	-0.4%	-0.3%	-0.3%	0.6%	-0.4%	-0.4%	-0.4%	-0.3%	-0.3%	-0.3%	-0.3%	-0.4%	-0.4%	-0.1%	0.0%	-0.2%	-0.4%	-0.4%	0.1%
	unifor..	ω	67%	63%	66%	37%	37%	66%	66%	63%	64%	31%	65%	67%	67%	64%	62%	66%	63%	67%	67%	55%	53%	60%	66%	67%	47%
		Δϕ	-0.6%	-0.5%	-0.6%	0.4%	0.4%	-0.7%	-0.6%	-0.5%	-0.4%	0.6%	-0.6%	-0.7%	-0.6%	-0.4%	-0.4%	-0.6%	-0.6%	-0.7%	-0.6%	-0.2%	-0.1%	-0.4%	-0.6%	-0.6%	0.1%
α = 50%	logno..	ω	61%	61%	62%	61%	28%	37%	59%	61%	28%	28%	55%	36%	62%	61%	28%	4%	20%	57%	62%	28%	3%	48%	32%	62%	28%
		Δϕ	-0.2%	-0.2%	-0.2%	-0.2%	0.2%	0.1%	-0.2%	-0.2%	0.4%	0.2%	-0.1%	0.1%	-0.2%	-0.1%	0.2%	1.2%	0.3%	-0.2%	-0.2%	0.2%	1.1%	-0.1%	0.1%	-0.2%	0.2%
	trian..	ω	66%	66%	66%	60%	35%	55%	65%	63%	36%	35%	62%	56%	65%	59%	35%	23%	47%	64%	65%	35%	20%	60%	54%	65%	35%
		Δϕ	-0.5%	-0.5%	-0.5%	-0.2%	0.3%	-0.3%	-0.5%	-0.4%	0.4%	0.3%	-0.4%	-0.3%	-0.5%	-0.2%	0.3%	0.9%	0.0%	-0.5%	-0.5%	0.3%	1.0%	-0.4%	-0.3%	-0.5%	0.3%
	unifor..	ω	67%	68%	67%	57%	36%	60%	66%	63%	40%	36%	62%	62%	67%	58%	36%	34%	55%	67%	67%	36%	30%	64%	60%	67%	36%
		Δϕ	-0.8%	-0.8%	-0.7%	-0.3%	0.4%	-0.5%	-0.8%	-0.5%	0.4%	0.4%	-0.5%	-0.6%	-0.7%	-0.3%	0.4%	0.6%	-0.4%	-0.9%	-0.7%	0.4%	0.9%	-0.6%	-0.6%	-0.7%	0.4%

Table 3.6 – Simulation results for instance #8, $n = 150$, for different degrees of uncertainty $\alpha \in \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Comparison is based on two measures: (i) $\omega(\Gamma)$ is the % of simulated scenarios (over a total of 10,000) where 2RPFS(Γ) obtained smaller *makespan* cost when compared to Det(P=); (ii) $\Delta\Phi(\Gamma) = Avg_{\lambda \in \mathbb{S}}\left[\frac{\varphi(\sigma^*_\Gamma, P^\lambda) - \varphi(\bar{\sigma}^*, P^\lambda)}{\varphi(\bar{\sigma}^*, P^\lambda)}\right]$ is the average relative cost difference between 2RPFS(Γ) and Det(P=), given all simulated scenarios λ.

Based on the simulation framework presented in Section 3.6.3, we close this section with a further analysis of the actual cost overhead of robust solutions in the long run. Two performance measures are calculated for each variability level α, as shown in Table 3.6. The obtained results show that, for different protection levels Γ, several solutions present two important characteristics: (i) high proportion of cheapest solutions ($\omega(\Gamma) > 50\%$), and (ii) smaller expected costs, i.e., negative relative cost difference $\Delta\Phi(\Gamma)$. Overall, 2RPFS provides a pool of robust schedules decision-makers can choose based on their risk preferences.

3.7 The m-machine Robust Permutation Flow Shop Problem (Rob-PFSP)

In this section, we extend the robust permutation flow shop problem, generalizing it to an arbitrary number of machines. We start by providing a formal definition for the makespan-objective m-machine Robust Flow Shop Problem (Section 3.7.1), followed by a description of the underlying budgeted uncertainty set (Section 3.7.2). Finally, two robust counterpart formulations are proposed (Section 3.7.3), based on well-known Mixed-Integer Linear Programming (MILP) formulations for the deterministic problem.

In the subsequent sections, we will develop and test two solution procedures for this problem: an exact C&CG method, used as baseline, followed by a GRASP metaheuristic. Both procedures make use of a dynamic programming algorithm which, given a scheduling σ and a protection level Γ, determines the worst-case scenario.

3.7.1 Problem statement

The following problem statement generalizes the 2RPFS definition from Section 3.4.1 to the m-machine case. Assume the matrix of individual processing times $\mathbf{P} = \{p_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$ contains uncertain data. A scenario λ is defined as a realization of uncertainty and, for each λ, there is a unique matrix of processing times denoted as $\mathbf{P}^\lambda = \{p^\lambda_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$. Let Λ be the set of all possible scenarios λ. Whenever a matrix of processing times $\mathbf{P}^\lambda$ is known, an instance of the deterministic PFSP is defined.

Let $\varphi(\sigma, \mathbf{P}^\lambda)$ be the *makespan* of a sequence $\sigma \in \Sigma$ given a scenario $\lambda \in \Lambda$. The objective of the m-machine Robust PFS is to find a job permutation $\sigma \in \Sigma$ that *minimizes* the *maximum* **makespan** over all scenarios $\lambda \in \Lambda$:

$$\textbf{RPFS:} \quad \min_{\sigma\in\Sigma} \max_{\lambda\in\Lambda} \{\varphi(\sigma, \mathbf{P}^\lambda)\} \tag{3.53}$$

For any sequence $\sigma \in \Sigma$, the value

$$\mathcal{Z}(\sigma) := \max_{\lambda\in\Lambda} \{\varphi(\sigma, \mathbf{P}^\lambda)\} \tag{3.54}$$

is called the *worst-case makespan* or *robust cost* for σ. The maximizer in (3.54) is called a worst-case scenario for σ.

3.7.2 Generalizing the budget uncertainty set to the m-machine problem

Similarly to Section 3.4.2, consider the interval approach for representing uncertain values. Two positive processing time matrices $\overline{P}$ and $\widehat{P}$ represent the nominal value and the maximum allowed deviation of P, respectively. To apply budgeted uncertainty to the m-machine problem, we introduce the budget parameter $\Gamma : 0 \leq \Gamma \leq mn$, which denotes the maximum number of operations whose uncertain processing times can reach their worst-case values. The *budgeted uncertainty set* of operations (processing time), denoted as $\mathcal{U}^{\Gamma}$, can be defined as follows:

$$\mathcal{U}^{\Gamma} = \left\{ \mathbf{P} = \{p_{r,i}\} : p_{r,i} = \overline{p}_{r,i} + \delta_{r,i}\, \widehat{p}_{r,i},\ \delta_{r,i} \in \{0,1\}, \forall r \in \mathbb{M}, \forall i \in \mathbb{J} : \sum_{r=1}^{m} \sum_{i=1}^{n} \delta_{r,i} \leq \Gamma \right\}, \quad (3.55)$$

Given uncertainty set $\mathcal{U}^{\Gamma}$, a scenario λ is described by one of the infinite matrices in this set. For a given operation $O_{r,i}$ concerning the execution of job i on machine r, let $\delta^{\lambda}_{r,i}$ be the value defining the deviation of its processing time, i.e., $p^{\lambda}_{r,i} = \overline{p}_{r,i} + \delta^{\lambda}_{r,i}\widehat{p}^{\lambda}_{r,i}$. Therefore, the total number of operations whose processing time can deviate to its maximum value is limited to Γ.

As mentioned before, the main advantage of applying budgeted uncertainty sets is the ability to model the risk-averseness of the decision maker by varying Γ. The idea is that an event where all uncertain parameters $p_{r,i}$ reach their worst-case values at the same time has a very low probability of happening [16]. In particular, higher values of Γ lead to larger uncertainty sets and thus more conservative solutions. When $\Gamma = 0$, the problem is equivalent to the nominal problem, i.e., the deterministic PFSP. If $\Gamma = mn$, we obtain the box uncertainty set [128]. For a given value of Γ, there are $\binom{mn}{\Gamma}$ possible worst-case scenarios, given the budgeted uncertainty set $\mathcal{U}^{\Gamma}$.

3.7.3 Robust counterparts

Based on existing PFSP MILP models, we developed two robust counterparts for the m-machine PFSP, based on Wilson [149] and Wagner [129] formulations. Both models rely on assignment constraints in order to find the position occupied by each job in the schedule. Certain parts of the models described below are similar to the ones introduced in Section 3.4.3, but adapted to the m-machine case. For completeness, we have chosen to repeat the introductory explanation of both models. For more details on the rationale behind each deterministic PFSP model, including illustrative diagrams, we refer the reader to [135].

3.7.3.1 Robust Counterpart for Wilson PFS Model

[149] proposed a MILP model for the *makespan*-minimizing flow shop scheduling problem, by applying sets of inequality constraints, based on the start time variables, of each job on each machine. In this work, we derived a two-stage robust counterpart of his model, with the following decision variables:

$Z_{i,j} =$	$\begin{cases} 1, \text{ if } \sigma(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma), \\ 0, \text{ otherwise.} \end{cases}$
$B^{\lambda}_{r,j} =$	start time of job $\sigma(j)$ (in position j) on machine M_r given scenario λ.

Based on the above definitions, variables $Z_{i,j}$ are in the first stage, and variables $B^{\lambda}_{r,j}$ are in the second stage of this robust counterpart. The two-stage robust-counterpart of Wilson model for the RPFS can be formulated as follows:

$$\text{Min } y \tag{3.56}$$

$$\text{st } B^{\lambda}_{m,n} + \sum_{i=1}^{n} \left(\overline{p}_{m,i} + \widehat{p}_{m,i} \delta^{\lambda}_{m,i} \right) Z_{i,n} \leq y, \qquad \lambda \in \Lambda, \tag{3.57}$$

$$B^{\lambda}_{1,1} = 0, \qquad \lambda \in \Lambda, \tag{3.58}$$

$$B^{\lambda}_{1,j} + \sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i} \delta^{\lambda}_{1,i} \right) Z_{i,j} = B^{\lambda}_{1,j+1}, \qquad j = 1, \ldots, n-1, \lambda \in \Lambda, \tag{3.59}$$

$$B^{\lambda}_{r,1} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i} \delta^{\lambda}_{r,i} \right) Z_{i,1} = B^{\lambda}_{r+1,1}, \qquad r = 1, \ldots, m-1, \lambda \in \Lambda, \tag{3.60}$$

$$B^{\lambda}_{r,j} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i} \delta^{\lambda}_{r,i} \right) Z_{i,j} \leq B^{\lambda}_{r+1,j}, \; r = 1, \ldots, m-1, j = 2, \ldots, n, \lambda \in \Lambda, \tag{3.61}$$

$$B^{\lambda}_{r,j} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i} \delta^{\lambda}_{r,i} \right) Z_{i,j} \leq B^{\lambda}_{r,j+1}, \; r = 2, \ldots, m, j = 1, \ldots, n-1, \lambda \in \Lambda, \tag{3.62}$$

$$\sum_{i=1}^{n} Z_{i,j} = 1, \qquad j = 1, \ldots, n, \tag{3.63}$$

$$\sum_{j=1}^{n} Z_{i,j} = 1, \qquad i = 1, \ldots, n, \tag{3.64}$$

$$Z_{i,j} \in \{0, 1\}, \qquad i, j = 1, \ldots, n, \tag{3.65}$$

$$B^{\lambda}_{r,j} \geq 0, \qquad r = 1, \ldots, m, j = 1, \ldots, n, \lambda \in \Lambda, \tag{3.66}$$

$$y \geq 0. \tag{3.67}$$

The objective function (3.56) and constraint (3.57) state that this formulation aims to find a robust schedule for the processing of n jobs that minimizes the *makespan* of the worst-case scenario, among all possible scenarios $\lambda \in \Lambda$. Constraints (3.58)-(3.62) guarantee that the robust schedule is feasible and that start time variables are appropriately calculated, for each scenario λ. Constraints (3.63) and (3.64) are the classical assignment constraints, ensuring, respectively, that each job is assigned to one and only one sequence position, and that each sequence position is filled by one and only one job. Finally, constraints (3.65)-(3.67) define the domain of the variables.

3.7.3.2 Robust Counterpart for Wagner PFS Model

[142] proposed an all-integer programming model for a three-machine deterministic flow shop, later extended to a m-machine MILP model by [129], and commonly named in the literature as *Wagner model*. In our research, we derived a two-stage robust counterpart of the Wagner model,

with the following decision variables:

$Z_{i,j} =$	$\begin{cases} 1, \text{ if } \sigma(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma), \\ 0, \text{ otherwise.} \end{cases}$
$X^{\lambda}_{r,j} =$	idle time on machine M_r before the start of job in sequence position j given scenario λ.
$Y^{\lambda}_{r,j} =$	idle time of job in sequence position j after it finishes processing on machine M_r given scenario λ.

Based on the above definitions, variables $Z_{i,j}$ are in the first stage, and variables $X^{\lambda}_{r,j}$ and $Y^{\lambda}_{r,j}$ are in the second stage of this robust counterpart. The Wagner model for the RPFS can be formulated as follows:

$$\text{Min } y \tag{3.68}$$

$$\text{st } \sum_{i=1}^{n} \left(\overline{p}_{m,i} + \widehat{p}_{m,i}\delta^{\lambda}_{m,i}\right) + \sum_{p=1}^{n} X^{\lambda}_{m,p} \leq y, \qquad \lambda \in \Lambda, \tag{3.69}$$

$$X^{\lambda}_{1,1} = 0, \qquad \lambda \in \Lambda, \tag{3.70}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,j+1} + X^{\lambda}_{r,j+1} + Y^{\lambda}_{r,j+1} = \sum_{i=1}^{n} \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta^{\lambda}_{r+1,i}\right) Z_{i,j} + X^{\lambda}_{r+1,j+1} + Y^{\lambda}_{r,j}, \qquad r = 1,\ldots,m-1, j = 1,\ldots,n-1, \lambda \in \Lambda, \tag{3.71}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,1} + X^{\lambda}_{r,1} + Y^{\lambda}_{r,1} = X^{\lambda}_{r+1,1}, \qquad r = 1,\ldots,m-1, \lambda \in \Lambda, \tag{3.72}$$

$$\sum_{i=1}^{n} Z_{i,j} = 1, \qquad j = 1,\ldots,n, \tag{3.73}$$

$$\sum_{j=1}^{n} Z_{i,j} = 1, \qquad i = 1,\ldots,n, \tag{3.74}$$

$$Z_{i,j} \in \{0,1\}, \qquad i,j = 1,\ldots,n, \tag{3.75}$$

$$X^{\lambda}_{r,j} \geq 0, Y^{\lambda}_{r,j} \geq 0, \qquad r = 1,\ldots,m, j = 1,\ldots,n, \lambda \in \Lambda, \tag{3.76}$$

$$y \geq 0. \tag{3.77}$$

The objective function (3.68) and constraint (3.69) state that this formulation aims to find a robust schedule for the processing of n jobs that minimizes the *makespan* of the worst-case scenario, among all possible scenarios $\lambda \in \Lambda$. Constraints (3.70)-(3.72) guarantee that the robust schedule is feasible and that idle time variables are appropriately calculated, for each scenario λ. Constraints (3.73) and (3.74) are as defined in Wilson formulation. Finally, constraints (3.75)-(3.77) define the domain of the variables.

Solving the aforementioned models for all possible combinations of $\lambda \in \Lambda$ is unrealistic. Therefore, in the next subsection, we will present an algorithm capable of obtaining optimal results for the m-machine RPFS with a subset of these combinations.

3.8 Solving Rob-PFSP with an exact C&CG method

Given that the uncertainty set $\mathcal{U}^{\Gamma}$ defined in Section 3.7.2 is discrete and finite, it is possible to obtain a solution to the RPFS problem by choosing an appropriate Robust Counterpart formulation and enumerating all variables and constraints for each scenario λ in the set Λ. This possibility, as we can expect, is unrealistic.

As an alternative approach, the C&CG framework [162] generates only a subset of scenarios $\Theta = \{\lambda_1, \ldots, \lambda_v\} \subseteq \Lambda$. For its application, the problem has to be formulated in a master-subproblem

framework. Solutions are then obtained iteratively, with each iteration generating one or more scenarios $\lambda_v \in \Theta$, obtained by solving a worst-case subproblem (SP).

In this section, we apply the same C&CG algorithm presented in Section 3.5, but this time with new Robust Counterparts and a new worst-case procedure for the m-machine robust problem. Using the RC formulations defined in Section 3.7.3, we derive two solution approaches, in which the Master Problem (MP) is based on either the Wilson or Wagner formulations.

Additionally, we define the worst-case scenario procedure, based on dynamic programming, for the solution of the subproblem (SP), under the budgeted uncertainty set $\mathcal{U}^{\Gamma}$, for a specific sequence of jobs $\sigma = \{\sigma(j), j = 1, \ldots, n\}$. By extending the notation of Equation (3.54), given a sequence σ and a protection level Γ, its *worst-case makespan* or *robust cost*, $\mathcal{Z}(\sigma, \Gamma)$ is given by:

$$\mathcal{Z}(\sigma, \Gamma) := \max_{\mathbf{P}^{\lambda} \in \mathcal{U}^{\Gamma}} \{\varphi(\sigma, \mathbf{P}^{\lambda})\} \tag{3.78}$$

For the sake of simplicity, we assume that budget parameter Γ is a non-negative integer. Based on this assumption, statement (3.78) reflects a problem with a convex function being maximized over a polytope defined by uncertainty set $\mathcal{U}^{\Gamma}$. Thus, in order to obtain the worst-case realization of uncertainty, only specific realizations of $\mathcal{U}^{\Gamma}$ are needed, namely the extreme points of the polytope. For each machine r and job i, the set of extreme point scenarios is characterized either by the values $\overline{p}_{r,i}$ or $\overline{p}_{r,i} + \hat{p}_{r,i}$. Any worst-case realization will use as much budget of uncertainty as possible. Therefore we can expect that, for the optimal solution of (3.78), with worst-case scenario λ^*, $\sum_{r=1}^{m} \sum_{i=1}^{n} \frac{|p_{r,i}^{\lambda^*} - \overline{p}_{r,i}|}{\hat{p}_{r,i}} = \Gamma$.

The following worst-case method is a generalization of the dynamic programming proposed in Section 3.5.2. The novelty of this new method involves the extension for an arbitrary number of machines m, and the use of a single budget parameter Γ, which limits the number of operations with deviated processing times. The complexity of the algorithm is $\mathcal{O}(mn^2)$. Given $1 \leq r \leq m,\ 1 \leq k \leq n$, and $0 \leq \gamma \leq n$, let us define a value function $\alpha(r, k, \gamma)$ as the optimal value of the restricted separation problem for machine r and job positions $\{1, \ldots, k\}$, when at most γ operations are using their maximum processing time. The optimal value of the problem is then defined by $\mathcal{Z}(\sigma) = \alpha(m, n, \Gamma)$.

The value-function is defined by the recursion:

$$\alpha(1, k, \gamma) = \max\Big[\overline{p}_{1,\sigma(k)} + \alpha(1,\ k-1,\ \gamma),\ \overline{p}_{1,\sigma(k)} + \widehat{p}_{1,\sigma(k)} + \alpha(1,\ k-1,\ \gamma-1)\Big],$$
$$for\ 1 \leq k \leq n,\ 0 \leq \gamma \leq \Gamma,$$
$$\alpha(r, k, \gamma) = \max\Big[\overline{p}_{r,\sigma(k)} + \max[\alpha(r,\ k-1,\ \gamma),\ \alpha(r-1,\ k,\ \gamma)]\ ,$$
$$\overline{p}_{r,\sigma(k)} + \widehat{p}_{r,\sigma(k)} + \max[\alpha(r,\ k-1,\ \gamma-1),\ \alpha(r-1,\ k,\ \gamma-1)]\Big],$$
$$for\ 2 \leq r \leq m,\ 1 \leq k \leq n,\ 0 \leq \gamma \leq \Gamma,$$

and the following initialization values:

$$\alpha(r, k, \gamma) = -\infty\ for\ 1 \leq r \leq m,\ 0 \leq k \leq n,\ \gamma < 0,$$
$$\alpha(r, 0, \gamma) = 0\ for\ 1 \leq r \leq m, \text{ and } 0 \leq \gamma \leq \Gamma,$$
$$\alpha(r, k, 0) = \overline{p}_{r,\sigma(k)} + \max[\alpha(r,\ k-1,\ 0),\ \alpha(r-1,\ k,\ 0)],\ 1 \leq r \leq m,\ 1 \leq k \leq n.$$

If $r = 1$, the first maximizer argument accounts for the case when there is no delay of execution regarding job $\sigma(k)$ on the first machine, while the second expression handles the case where a delay occurs. Similarly, for $r \geq 2$, we take the maximum of two cases: with or without delay when executing job $\sigma(k)$ on machine r. However, the job start time has to be computed as the maximum between the previous job's $\sigma(k-1)$ worst-case completion time on the same machine M_r, and the worst-case completion time of the same job $\sigma(k)$ on the previous machine M_{r-1}, taking into account the current budget γ.

3.9 Solving Rob-PFSP with the GRASP metaheuristic

Due to the NP-hardness of the regular PFSP with three or more machines, only relatively small instances can be solved by exact methods. To solve larger problems, several approximate methods have been developed in the literature. For a comparison of the existing heuristics and metaheuristics for the *makespan* objective, we refer the reader to [39]. Since the robust PFSP is at least as difficult as regular PFSP, heuristics are very much needed. In this section, to solve the RPFS Problem, we apply the GRASP metaheuristic [37], which has been successfully applied in other scheduling problems [17, 47]. The main idea is to reuse the worst-case calculation procedure, defined in Section 3.8, as the objective function of GRASP.

GRASP is a multi-start metaheuristic in which each iteration consists basically of two phases: construction and local search [37]. The solutions generated by a GRASP construction procedure are not guaranteed to be locally optimal with respect to simple neighborhood definitions. Hence, it is almost always beneficial to apply a local search to attempt to improve each constructed solution.

Our $RGRASP$ is described in Algorithm 5. Instead of limiting the number of iterations, the only adopted stopping criterion is the time limit. The first task in each iteration of $RGRASP$ is the construction of an initial solution in a greedy randomized fashion. In this phase, a feasible solution is progressively constructed with the addition of solution elements, one at a time. Choosing the next element to be added is determined using a greedy function, which measures the impact of incorporating each element in the partial solution under construction. Besides this greedy evaluation, the constructive phase of GRASP also includes a probabilistic component. Each construction step randomly selects an element among the most promising candidates, although not necessarily the top one, thus allowing the generation of different initial solutions, at each iteration of GRASP. In other words, the best rated elements (still unassigned to the current solution) are stored in a Restricted Candidate List (RCL), from which the next element to be added to the solution is taken.

The $ConstructivePhase$ procedure is described in Algorithm 3. For partial solution evaluation, we have applied a robust cost (minimization) gain function, suitable for Robust Optimization problems. In order to formalize it, we need some additional notation. Let $\sigma_P = \{j_1, j_2, \ldots, j_k\}$ denote a *partial job permutation* (i.e., a sequence of jobs of size $|\sigma_P| < n$). We define below a *function*

$g : \sigma_P \rightarrow \mathbb{R}$, which will measure the impact on the robust cost $\mathcal{Z}(\sigma_P)$ of inserting a new job $j \in \mathbb{J} \backslash \sigma_P$ in the partial permutation σ_P.

$$g(\sigma_P) = \min_{j \in \mathbb{J} \backslash \sigma_P} \mathcal{Z}(\sigma_P \cup \{j\}). \tag{3.79}$$

This minimization function compares the cost of inserting each unassigned job j into the partial permutation σ_P. Note that a job with a low gain function value will probably contribute less to the robust cost if we add it to the partial permutation σ_P.

Algorithm 3: *ConstructivePhase*

```
1 Input: P̄, P̂ and α
2 Output: permutation σ
3 σ = ∅; L_g = Order(J)
4 while L_g ≠ ∅ do
5     Choose job j randomly among the first ⌊α. | L_g |⌋ elements of L_g
6     Update σ = σ ∪ {j}
7     L_g = L_g − {j};  Re-order(L_g)
8 return σ
```

Algorithm 4: *VariableNeighborhoodDescent*

```
1  Input: P̄, P̂ and permutation σ
2  Output: permutation σ
3  r = 1
4  while r ≤ vnd-size do
5      for σ̄ ∈ N_r(σ) do
6          if Z(σ̄) < Z(σ) then
7              r = 1
8              σ = σ̄
9      if Z(σ) has not improved then
10         r = r + 1
11 return σ
```

In this phase, the ordered set L_g is defined (line 3) as the set of jobs $\mathbb{J} \backslash \sigma_P$ ordered in increasing order of function g (Figure 3.4). At each iteration, in lines 4-7, we randomly choose a job j among the first $\lfloor \alpha . \mid L_g \mid \rfloor$ vertices in this set and add it to the partial permutation. This process is repeated until a full permutation σ is obtained.

Notice that this technique tends to generate different solutions every time the multi-start procedure (i.e., iteration) is run, which helps the GRASP algorithm to avoid getting trapped into a local minimum. Additionally, the α parameter provides a compromise between search exploitation (intensification using more greediness) and search exploration (diversification using more randomness).

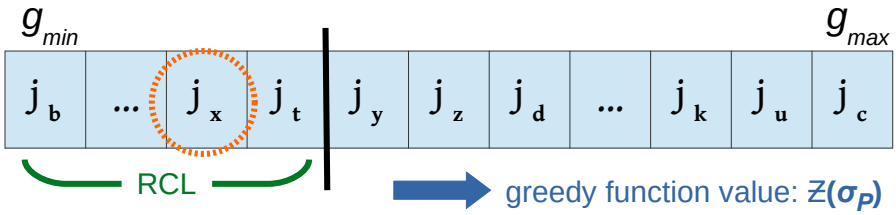

Figure 3.4 – Constructive phase.

Small values improve the average quality of the RCL list elements and encourage exploitation. When $\alpha = 0$, the constructive phase becomes a deterministic greedy algorithm, while an alpha value of one makes the algorithm completely random.

In this work, after extensive experiments, in order to determine the value of the α parameter, we opted for the so-called dynamic approach. In other words, the α value is randomly chosen from a uniform distribution in the range $[\beta_1, \beta_2]$, at each construction of a new solution. The values β_1 and β_2 then become parameters for the GRASP procedure.

Algorithm 5: *RGRASP*

```
1 Input: P̄, P̂ and α
2 Output: permutation σ*
3 σ* = ∅;  Z(σ*) = ∞
4 while time limit is not reached do
5     σ = ConstructivePhase(P̄, P̂, α)
6     σ = VariableNeighborhoodDescent(P̄, P̂, σ)
7     if Z(σ) < Z(σ*) then
8         σ* = σ
9 return σ*
```

There is no guarantee that the construction method returns a (locally) optimal solution with respect to some neighborhood. Therefore, the solution σ obtained in $ConstructivePhase$ can be improved by the local search procedure $VariableNeighborhoodDescent$ (Algorithm 4).

The Variable Neighborhood Search (VNS) method, proposed by Mladenović and Hansen [98], is a metaheuristic that explores distant neighborhoods of the current incumbent solution, and moves to a new solution if and only if an improvement is made. The main idea is to systematically explore differing neighborhood structures, with the goal of escaping from local minima. Within this work, we apply the Variable Neighborhood Descent (VND) search, a variant of VNS.

As illustrated in Figure 3.5, VND starts with the permutation provided by the construction phase and iteratively compares the incumbent value $\mathcal{Z}(\sigma)$ with the new value $\mathcal{Z}(\bar{\sigma})$ obtained in the $r - neighborhood$, denoted by $\mathcal{N}_r(\sigma)$ and defined as the family of all permutations obtained by

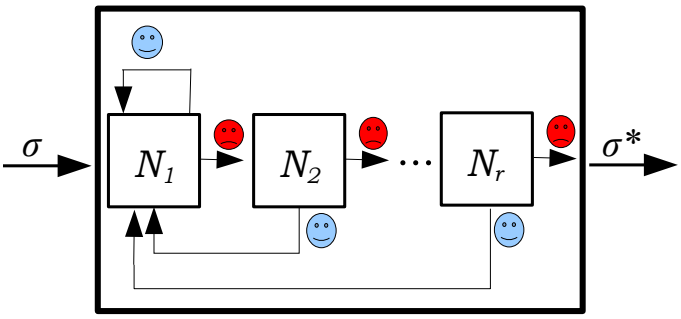

Figure 3.5 – Variable Neighborhood Descent procedure. N_1 stands for neighborhood #1, N_2 stands for neighborhood #2, and so on.

traversing the r-th neighborhood structure over σ. In this work, we employ a local search approach similar to the one used by [118], but with the inclusion of other structures, in addition to the insertion neighborhood, as seen on Table 3.7. If an improvement is obtained, r is returned to its initial value and the new incumbent updated (lines 7 and 8 in Algorithm 4). Otherwise, the next neighborhood is considered (line 10). The local search halts when no better partition is found in the most distant neighborhood of the current solution. Additionally, our implementation of the VND procedure includes a *Random-VND* (RNVD) parameter [125], allowing to shuffle the order according to which the neighborhood structures from Table 3.7 will be traversed by VND.

Remark that, in all employed solution neighborhoods, each evaluated solution is further improved by shifting all modified jobs to the left ("shift-to-left movement"). Moreover, since these structures are huge (great number of permutations), they are not traversed completely. The search in neighborhood #1 stops whenever it finds an improvement of the current solution (first improvement rule). In the other neighborhoods, only a single (random) movement is evaluated at each invocation.

#	Name and Description
1	**Move** Remove each job from its current position and insert it in all remaining positions.
2	**Random Swap** Select two jobs at random and swap their positions in the permutation.
3	**Random Move End** ($d = 2$) After removing d random jobs from the permutation (destruction), reinsert each one in the last d positions (construction).

Table 3.7 – Solution neighborhoods employed in the VND procedure, ordered by complexity. The *RVND* parameter allows to shuffle the order according to which these neighborhood structures are applied. In all employed solution neighborhoods, each evaluated solution is further improved by shifting all modified jobs to the left ("shift-to-left movement").

Similarly to [68], as the termination criteria for the multi-start procedure, *max-time* seconds is defined by the product of the number of jobs, n, the number of machines, m, and a time factor t = 0.03, such that: *max-time* $= n \times m \times 0.03$. This leads to very short computation times, ranging from only a few seconds to a maximum of 5 minutes for the largest instances.

3.10 Rob-PFSP experimental results

We conducted extensive experiments to assess the performance of both the exact and heuristic solution methods when solving Rob-PFSP, including the quality of the obtained heuristic schedules.

3.10.1 Test instances

In the PFSP literature, there exists a classical set of very well-known benchmarks commonly used to test new algorithmic approaches. However, except for the 2-machine instances proposed by [159], there are no standard benchmarks for the m-machine Rob-PFSP.

With this in mind, in order to verify the effectiveness of the proposed algorithms, our experiments were based on Ying instances (i), along with three new instance sets (ii, iii, iv) [1]:

(i) **Two-machine robust PFSP instances**, originally proposed by [159]. In his work, six groups of instances were created, each one with a different number of jobs $n \in \{10, 20, 50, 100, 150, 200\}$. The expected processing time $\overline{p}_{r,i}$ $(r = 1, 2;\ i = 1, \ldots, n)$ is an integer drawn from the uniform distribution [10, 50] and the largest processing time deviation was set as a ratio of the expected processing time (i.e., $\widehat{p}_{r,i} = \alpha \overline{p}_{r,i}$), where $\alpha \in \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Ten instances were generated for each combination of n and α, for a total of 300 test instances.

(ii) **Robust PFSP instances with 3, 4, 5, 10 and 15 machines**. Following the same instance generation algorithm of instance set (i) [159], we generated random instances with 10 jobs and 3, 4, 5, 10 and 15 machines, respectively. Ten instances were generated for each combination of m, n and α, for a total of 250 test instances.

(iii) **Taillard-based instances** [2]. In order to generalize the classical Taillard PFS instances [132] to the robust PFS problem, we assumed the provided processing times as nominal values ($\overline{p}_{r,i}$). Similarly to the instance generation algorithm of [159], the maximum processing time deviations were generated as a ratio of the expected processing time (i.e., $\widehat{p}_{r,i} = \alpha \overline{p}_{r,i}$), where $\alpha = \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Therefore, five robust instances were generated for each original Taillard PFSP instance, yielding a total of 500 test instances.

1. All test instances are available at https://github.com/levorato/RPFS_Cmax_Budget.

2. Taillard instances are grouped in 12 sets of 10 instances each, according to the number of jobs and the number of machines (i.e., Set 20 x 5, set 20 x 10, set 20 x 20, set 50 x 5, set 50 x 10, set 50 x 20, set 100 x 5, set 100 x 10, set 100 x 20, set 200 x 10, set 200 x 20, and set 500 x 20).

(iv) **Robust PFSP instances with random processing time deviations**. For each instance of the previous three sets with variability level $\alpha = 10\%$, we generated 4 new instances with distinct variability levels $\alpha_{r,i}$ for each operation $O_{r,i}$. First, we define a maximum variability level $\alpha^{max} \in \{30\%, 50\%, 100\%, 200\%\}$. Then, in each generated instance, the variability level $\alpha_{r,i}$ of each operation $O_{r,i}$ is drawn from a uniform distribution in the interval $[0, \alpha^{max})$. Therefore, the maximum processing time deviation of each operation equals $\widehat{p}_{r,i} = \alpha_{r,i}\overline{p}_{r,i}$. The idea behind this new set is to generate instances whose operation processing time deviation follow a completely random behavior, when compared to the previous sets. This way, we will be able to assess the impacts of such behavior on the solution method.

3.10.2 Implementation details and algorithm parameters

The C&CG algorithm was coded in Julia 1.6.0, and the IBM ILOG CPLEX solver 12.9.0 was used to solve the Mixed-integer programs. The GRASP algorithm was coded in C++ 11. All experiments were conducted on a workstation with an Intel Xeon® CPU E5640 @2.67GHz with 32 GB RAM, under Ubuntu 18.04 LTS.

In the Exact method, we used CPLEX (default parameters) to solve the MILP models with time limit of 7,200 s. Relative optimality gap tolerance was set to 10^{-6}. On the other hand, in the GRASP procedure, the time limit for the solution of each instance varies according to its size. In order to determine the best set of GRASP parameters, for each instance, Rob-PFSP was solved for all $\Gamma \in \{10\%, 20\%, \ldots, 100\%\}$. Moreover, all values in Table 3.8 were tested.

Only a subset of problem instances were used for GRASP parameter calibration. For this purpose, we randomly chose 20% of the instances from each group. Parameter tuning was performed with the irace package [90]. For each value of budget Γ, we ran irace twice in the following way. In a first run, irace generated the initial configurations by uniformly sampling the parameters space (Table 3.8). In the second run, the best configurations given at the end of the first run were used as the new set of initial configurations. The final parameter set used in the experiments was defined in the end of the second run, by choosing the best configuration returned by irace, among all budget values $\Gamma \in \{10\%, 20\%, \ldots, 100\%\}$. In this study, the number of algorithm executions per irace run was set to 10^4, yielding a total parametrization budget of 10^5 GRASP executions, considering all Γ values. The final overall time consumed in the GRASP configuration was five days wallclock time, using 8 computing cores. Apart from the previous description, the remaining irace default settings were used.

3.10.3 Comparative performance of the Robust Counterpart models

When assessing the performance of the exact solution method, we solved each problem instance with different values of the budget of uncertainty parameter Γ. Therefore we obtained solutions for

Parameter name	Tested values
random-VND	false, <u>true</u>
vnd-size	1, 2, <u>3</u>
first-improvement	false, <u>true</u>
time-factor	30, <u>300</u>
β_1	0.2, <u>0.4</u>, 0.6, 0.8, 1.0
β_2	0.2, 0.4, 0.6, 0.8, <u>1.0</u>

Table 3.8 – GRASP parameter values evaluated during calibration. The best performing parameter values are underlined.

each RC model by varying Γ according to ten ratios (10, 20, 30, 40, 50, 60, 70, 80, 90 and 100%) of operations with uncertain processing times.

We first present overall results in Table 3.9, by comparing the performance of Wagner and Wilson RC models, for each instance size. When using average, the results of all instances (even outliers) are taken into account. Standard deviation is also included as a secondary measure. Additionally, the average number of iterations and its standard deviation are listed. As we could expect, these results show that, as instance size grows, the models become harder to solve, as seen on the smaller percentage of solved instances and increased average execution time.

Regarding the hardest instances of the test-bed, namely Ying 15×5 and Taillard 20×5, Wagner-based C&CG is the method that solves the majority of instances within the time limit: 64.4% and 24.7%, respectively. It also presents the smallest average gap among instances not solved to optimality. Additionally, the *% Best Performance* measurement indicates that Wagner model tends to solve Taillard 20×5 instances faster than Wilson.

A further investigation, based on the α and α^{max} parameters, is portrayed in Tables 3.10 and 3.11. In this context, we explore solution statistics regarding Ying 15×5 and Taillard 20×5 instances, respectively. It is possible to note the decrease of model performance as α and α^{max} values increase. This can be observed in the *% Solved*, *Avg. % gap* and *Avg. time* rows, from columns $\alpha = 10\%$ until $\alpha = 50\%$, and from columns $\alpha^{max} = 30\%$ until $\alpha^{max} = 200\%$.

It is worth noting that, when solving the Taillard 20×5 instances, the hardest ones in this comparison, we perceive a drastic performance reduction of the C&CG algorithm. The percentage of solved instances drops to 25% when applying the best-performing Wagner model. Also, when analyzing the % gap of instances not solved to optimality, the average % gap is considerably higher (12%), as well as its standard deviation (19%). In these cases (75% of all tested instances), the C&CG algorithm was not able to obtain an optimal solution within the 2-hour time limit.

Having achieved the limits of the exact solution method, in the next subsection, we will analyse the performance and solution quality of the GRASP metaheuristic, using C&CG solutions as a baseline for comparison.

	Instance type / Instance size / Model											
	Ying											
	10x2		20x2		50x2		100x2		150x2		200x2	
Measure	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson
% Best Performance	59.5	49.0	61.1	40.3	70.2	29.8	75.0	25.0	76.1	23.8	76.6	23.3
% Solved	99.5	99.5	99.3	99.4	99.3	99.3	98.8	98.9	94.5	97.6	88.0	93.9
Avg. % gap	0.6	0.9	0.6	0.8	0.1	0.5	0.1	0.1	0.2	0.0	0.2	0.1
Std. dev. of % gap	0.6	1.1	0.5	1.4	0.1	0.9	0.1	0.1	0.7	0.0	0.5	0.1
Avg. iterations	15.3	11.4	12.9	8.6	9.4	6.4	9.4	6.5	10.8	7.2	15.3	10.1
Std. dev. of iterations	170.1	119.5	105.5	69.9	59.6	32.0	34.6	26.4	29.7	21.2	37.7	24.5
Avg. MP time	38.5	39.4	52.0	47.1	57.5	54.6	128.2	102.8	466.9	242.5	952.0	553.1
Avg. SP time	1.1	0.3	1.4	0.5	1.1	1.0	3.7	2.0	14.0	6.2	14.0	19.8
Avg. time	39.5	39.7	53.4	47.6	58.6	55.6	131.8	104.8	480.9	248.7	966.0	572.9
Std. dev. of time	522.8	522.9	600.9	572.8	611.6	596.7	867.2	761.2	1,665.7	1,140.9	2,300.1	1,777.1

	Ying								Taillard	
	10x3		10x4		10x5		15x5		20x5	
Measure	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson
% Best Performance	39.0	61.8	51.4	48.9	29.5	70.6	44.1	56.0	58.1	41.9
% Solved	97.6	99.6	97.3	96.8	94.3	93.9	64.4	63.6	24.7	18.3
Avg. % gap	1.3	1.1	1.1	1.3	1.4	1.3	2.8	3.3	11.5	13.9
Std. dev. of % gap	1.5	0.9	1.1	1.3	1.4	1.3	3.7	4.2	18.5	20.0
Avg. iterations	43.7	11.4	38.2	31.5	37.5	42.7	22.7	19.6	16.1	12.0
Std. dev. of iterations	233.3	69.2	184.0	130.3	130.8	153.2	27.0	25.9	25.6	15.8
Avg. MP time	182.6	31.6	310.5	305.1	599.2	607.9	2,235.2	2,151.5	2,617.6	2,660.9
Avg. SP time	3.3	5.7	5.6	4.8	46.0	23.3	1,301.1	1,357.7	2,872.8	2,616.5
Avg. time	185.9	37.3	316.1	309.9	645.2	631.2	3,536.3	3,509.2	5,490.4	5,277.3
Std. dev. of time	1,122.7	469.6	1,299.6	1,334.4	1,825.4	1,839.5	3,098.5	3,256.6	3,734.1	4,627.3

Table 3.9 – Robust PFSP C&CG performance comparison, for each instance size $n \times m$ and RC model. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; Avg. % Gap and Std. dev. of % Gap are the mean and standard deviation of the percentage gap of solutions from instances not solved to optimality; Avg. iterations and Std. dev. of iterations are the mean and standard deviation of the number of iterations performed; Avg. MP(SP) time is the average time (in seconds) to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

	Measure	α=10%	α=20%	α=30%	α=40%	α=50%	α^{max}=30%	α^{max}=50%	α^{max}=100%	α^{max}=200%
Wagner	% Best Performance	45.2	40.1	49.2	47.1	48.6	43.6	38.0	40.8	45.2
	% Solved	81.9	72.4	63.3	61.0	54.8	78.6	61.0	51.0	56.2
	Avg. % gap	0.5	1.1	1.6	2.2	2.6	0.7	1.8	2.9	8.8
	Std. dev. of % gap	0.4	0.8	1.3	1.7	2.2	0.5	1.5	2.4	6.3
	Avg. iterations	13.5	21.1	26.4	29.1	30.9	15.4	19.7	27.5	20.7
	Std. dev. of iterations	13.2	21.4	32.9	35.0	38.2	14.4	17.0	30.0	21.6
	Avg. MP time	1,556.4	2,025.9	2,433.6	2,624.7	2,736.5	1,518.8	2,481.0	2,555.3	2,184.7
	Avg. SP time	415.7	918.2	1,084.4	1,294.8	1,385.5	924.5	1,451.2	2,037.3	2,197.9
	Avg. time	1,972.1	2,944.1	3,517.9	3,919.4	4,122.0	2,443.3	3,932.2	4,592.6	4,382.6
	Std. dev. of time	2,706.2	2,986.5	3,061.7	3,092.4	3,074.7	2,838.9	2,985.9	3,021.7	3,087.8
Wilson	% Best Performance	55.4	59.9	50.8	52.9	52.4	56.4	62.0	59.2	54.8
	% Solved	84.3	74.8	65.7	59.1	51.0	76.2	61.0	52.9	47.6
	Avg. % gap	0.4	1.2	1.7	2.6	3.3	0.9	2.1	4.0	8.3
	Std. dev. of % gap	0.3	0.8	1.3	2.3	2.8	0.6	1.7	3.7	7.0
	Avg. iterations	15.7	17.4	21.7	24.2	23.7	15.2	17.3	24.3	16.7
	Std. dev. of iterations	37.2	16.5	21.3	23.0	22.6	34.6	14.1	32.8	17.2
	Avg. MP time	1,500.1	2,014.7	2,500.6	2,615.9	2,746.8	1,261.5	2,247.3	2,308.5	2,168.6
	Avg. SP time	310.9	690.0	967.8	1,273.0	1,550.8	1,237.8	1,470.3	2,148.4	2,570.2
	Avg. time	1,810.9	2,704.7	3,468.4	3,888.9	4,297.6	2,499.2	3,717.6	4,456.8	4,738.9
	Std. dev. of time	2,755.8	3,095.9	3,276.8	3,116.9	3,089.7	3,135.2	3,142.8	3,127.8	3,373.2

Table 3.10 – Robust PFSP C&CG performance comparison for Ying instance size 15×5, grouped by α and α^{max} values, and RC model. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; Avg. % Gap is the average percentage gap of solutions from instances not solved to optimality; Avg. iterations is the average number of iterations performed; Avg. time MP(SP) is the average time to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

	Measure	α=10%	α=20%	α=30%	α=40%	α=50%	α^{max}=30%	α^{max}=50%	α^{max}=100%	α^{max}=200%
Wagner	% Best Performance	64.4	70.0	62.5	61.1	43.8	55.0	50.0	52.4	43.5
	% Solved	39.5	21.0	11.9	10.0	10.5	41.4	34.0	29.1	25.7
	Avg. % gap	5.8	8.8	14.2	19.5	23.8	1.4	2.0	12.1	6.5
	Std. dev. of % gap	4.8	9.2	15.0	22.3	26.8	2.0	1.5	28.3	5.5
	Avg. iterations	8.0	9.7	10.4	11.7	12.4	16.3	21.3	25.6	30.0
	Std. dev. of iterations	9.5	11.2	12.6	14.7	16.0	17.2	29.6	39.5	43.3
	Avg. MP time	2,077.4	2,690.1	2,897.7	3,005.4	3,026.2	2,331.4	2,650.7	2,477.1	2,404.2
	Avg. SP time	1,193.9	2,097.5	2,185.6	2,233.7	2,391.2	3,640.5	3,683.3	4,120.6	4,359.2
	Avg. time	3,271.3	4,787.6	5,083.3	5,239.1	5,417.4	5,971.9	6,334.0	6,597.7	6,763.3
	Std. dev. of time	3,602.3	4,061.6	3,915.7	3,976.7	4,107.3	3,328.1	3,143.9	2,936.6	3,042.1
Wilson	% Best Performance	35.6	30.0	37.5	38.9	56.3	45.0	50.0	47.6	56.5
	% Solved	31.9	18.1	12.4	9.1	10.0	29.0	23.8	19.1	21.4
	Avg. % gap	5.4	11.1	14.3	20.3	26.8	1.3	3.0	15.9	8.9
	Std. dev. of % gap	4.7	9.6	14.1	21.4	26.9	1.0	4.7	33.2	15.4
	Avg. iterations	7.9	9.4	10.4	11.1	12.1	14.5	14.0	16.2	19.9
	Std. dev. of iterations	9.9	11.6	13.1	14.3	15.4	12.9	15.8	19.7	27.0
	Avg. MP time	2,073.1	2,644.9	2,878.2	2,957.3	2,923.9	2,972.7	2,396.9	2,489.0	2,438.1
	Avg. SP time	1,355.4	1,795.4	1,932.3	2,012.1	1,863.9	3,653.8	4,453.4	4,575.0	5,366.9
	Avg. time	3,428.5	4,440.4	4,810.5	4,969.3	4,787.8	6,626.5	6,850.4	7,064.0	7,805.0
	Std. dev. of time	3,837.8	4,055.5	4,147.6	4,277.9	4,071.9	3,034.7	3,117.7	2,877.0	8,583.9

Table 3.11 – Robust PFSP C&CG performance comparison for Taillard instance size 20×5, grouped by α and α^{max} values, and RC model. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; Avg. % Gap is the average percentage gap of solutions from instances not solved to optimality; Avg. iterations is the average number of iterations performed; Avg. time MP(SP) is the average time to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

3.10.4 Comparative performance of GRASP and C&CG

Due to its stochastic nature, for each tested instance, 50 independent GRASP executions (replicas) were run. Each replica was run for a maximum time $t_{max} = 0.03 \times n \times m$ seconds, where n is the number of jobs and m is the number of machines. For the smallest instances $t_{max} = 1$ second, while for the largest ones, $t_{max} = 300$ seconds. A summary of GRASP execution statistics is displayed in Table 3.12, in terms of run time and number of iterations performed. On average, the algorithm takes no more than three minutes to obtain a solution to the problem.

Instance type	Instance size	Median time	Avg. time	Std. dev. of time	Median iterations	Avg. iterations	Std. dev. of iterations
Ying	10x2	6	6	11	8,423	8,708	3,098
	20x2	12	12	9	1,274	1,591	677
	50x2	30	30	8	87	120	53
	100x2	61	61	0	19	18	3
	150x2	94	93	1	5	5	1
	200x2	120	124	44	1	1	1
	10x3	9	9	0	5,173	6,251	2,834
	10x4	12	12	9	3,234	3,205	408
	10x5	15	15	20	2,074	2,152	438
	15x5	23	23	1	685	681	157
Taillard	20x5	30	30	7	247	244	66
	20x10	60	60	28	106	104	23
	50x5	76	76	1	17	17	7
	20x20	120	120	4	53	51	13
	50x10	153	153	2	6	6	2

Table 3.12 – GRASP solution statistics, grouped by instance type and instance size. Median time, Avg. time and Std. dev. of time are the median, mean and standard deviation of solution time (in seconds), respectively. Median iterations, Avg. iterations and Std. dev. of iterations are the median, mean and standard deviation of the number of GRASP iterations, respectively.

We now compare the solutions obtained by the GRASP and C&CG robust methods when solving the same set of instances. Each solution provides an upper bound for the optimal robust cost of the problem. Let UB_{CCG} and UB_{GRASP} be the upper bounds obtained by the C&CG and GRASP methods, respectively. Also, let LB_{CCG} be the lower bound obtained by the C&CG method. Whenever $LB_{CCG} = UB_{CCG}$ or $LB_{CCG} = UB_{GRASP}$, we have an optimal solution to the problem. To assess the quality of the heuristic solutions obtained with GRASP, we compare their solution value, denoted as UB_{GRASP}, with both bounds provided by the C&CG method. We define $gap_{LB}\%$ and $gap_{UB}\%$ as:

$$gap_{LB}\% = \frac{UB_{GRASP} - LB_{CCG}}{LB_{CCG}} \times 100, \tag{3.80}$$

$$gap_{UB}\% = \frac{UB_{GRASP} - UB_{CCG}}{UB_{CCG}} \times 100. \tag{3.81}$$

For each instance group, Table 3.13 shows the percentage of instances which were solved optimally by the GRASP metaheuristic (i.e., $gap_{LB}\% = 0$, considering the best solution found after 50 independent executions) and by the C&CG exact method within 7,200 seconds. The determination

of optimal solutions is based on the upper bound of each solution method, relative to the lower bound obtained by the C&CG method. GRASP was able to optimally solve all instances, except for 200×2 and 15×5 and 20×5 groups. Even so, regarding the 15×5 and 20×5 groups, GRASP solved far more instances to optimality than C&CG, reaching 97% and 68%, respectively.

Additionally, we deepen the analysis of the percentage of instances optimally solved, grouping by alpha and $\Gamma\%$ values, on Tables 3.14 and 3.15, respectively. Our evaluation is restricted to the three instance groups for which GRASP did not obtain 100% optimal solutions. By observing Table 3.14, it is possible to note the decrease of GRASP performance as α and α^{max} values increase. Also, as the $\Gamma\%$ parameter values grows (Table 3.15), so does the difficulty of obtaining optimal solutions, as the number of optimally solved instances decreases.

	Instance type / Instance size										
	Ying										Taillard
Solution Method	10x2	20x2	50x2	100x2	150x2	200x2	10x3	10x4	10x5	15x5	20x5
C&CG	99.5	99.3	99.3	98.8	94.5	88.0	97.6	97.3	94.3	64.4	22.3
GRASP	100.0	100.0	100.0	100.0	100.0	85.5	100.0	100.0	100.0	96.9	68.3

Table 3.13 – Percentage of instances solved to optimality by C&CG and GRASP solution methods, grouped by instance type and instance size.

		Alpha								
Instance Type	Instance Size	α=10%	α=20%	α=30%	α=40%	α=50%	α^{max}=30%	α^{max}=50%	α^{max}=100%	α^{max}=200%
Ying	10x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	20x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	50x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	100x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	150x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	200x2	91.9	89.1	81.9	98.3	96.4	80.4	80.2	88.0	61.8
	10x3	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	10x4	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	10x5	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	15x5	97.6	97.2	97.2	97.8	94.9	95.2	95.5	99.5	97.0
Taillard	20x5	57.4	58.8	59.3	64.1	58.2	68.4	83.3	94.3	96.7

Table 3.14 – Percentage of instances solved to optimality by GRASP, grouped by instance type, instance size and instance alpha.

A further analysis was conducted to examine the gaps in solution values obtained by GRASP. In Figure 3.6, for each instance size group, we present box plots depicting $gap_{LB}\%$ (left plot) and $gap_{UB}\%$ (right plot) values obtained when comparing both lower and upper bounds of C&CG against GRASP solution values. All gap values were obtained from 50 independent executions of GRASP, including all instances of the corresponding size and all $\Gamma\%$ parameters values used to test the algorithm. Observe that, for all instance groups except 20×5, the majority of the obtained solutions present zero or near zero optimality gap $gap_{LB}\%$, i.e., all $Q1 - 1.5$ IQR[3] and $Q3 + 1.5$ IQR box plot values tend to zero. Also, in many occasions, $gap_{UB}\%$ values reveal that GRASP achieved

3. IQR is defined as the difference between the 75th (Q3) and 25th (Q1) percentiles of the data.

		Γ %									
Instance Type	Instance Size	10	20	30	40	50	60	70	80	90	100
Ying	10x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	20x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	50x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	100x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	150x2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	200x2	91.0	89.0	81.9	88.9	82.4	84.4	83.1	82.3	81.5	80.3
	10x3	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	10x4	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	10x5	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	15x5	97.6	99.3	97.8	98.0	97.2	96.8	96.7	96.4	95.6	95.4
Taillard	20x5	83.4	73.5	71.4	70.4	71.4	69.8	69.3	67.0	68.1	63.4

Table 3.15 – Percentage of instances solved to optimality by GRASP, grouped by instance type, instance size and $\Gamma\%$.

improved solution values (i.e., negative gaps) when compared to C&CG. In particular, for the case of 20×5 instances not solved to optimality, 75% of the observed GRASP optimality gaps ($gap_{LB}\%$) are smaller than 10%. Another interesting observation is that, for each solution group, composed by instance and $\Gamma\%$ parameter value, the observed deviation in GRASP solution values, after 50 independent executions, is minimal. This is a strong indication of the robustness of the implemented metaheuristic.

Finally, we drill down to instance groups 15×5 and 20×5 in order to analyze the gap between the solutions obtained by C&CG and GRASP ($gap_{UB}\%$). Figure 3.7 helps visualize the gap distribution of 15×5 instances, depending on either the alpha value, which specifies the maximum degree of processing time deviation, or the Γ values, used as input to determine the level of solution robustness. Observe that, in the experiments, the obtained gaps are always smaller than 2%, and solution improvements are frequent, especially when solving the α^{max} instances from group (iv) or when $\Gamma\% \leq 30$.

Similarly, Figure 3.8 shows the $gap_{UB}\%$ distribution for Taillard 20×5 instances. The box plot grouped by alpha values reveals significant solution improvements when $\alpha^{max} \geq 50\%$, and very small positive gaps ($gap_{UB}\% \leq 2$), when they occur. The box plot on the right, grouped by $\Gamma\%$ values, also reveals very small positive gaps ($gap_{UB}\% \leq 2$), along with negative gaps of up to -9%.

We also ran the GRASP method to obtain solutions for the remaining Taillard-based instances proposed in Section 3.10.1. Even though we were unable to evaluate their solution quality (due to problem-size limitations of the baseline C&CG method), we published the best solution found for each instance and $\Gamma\%$ value, along with the instance file set at https://github.com/levorato/RPFS_Cmax_Budget.

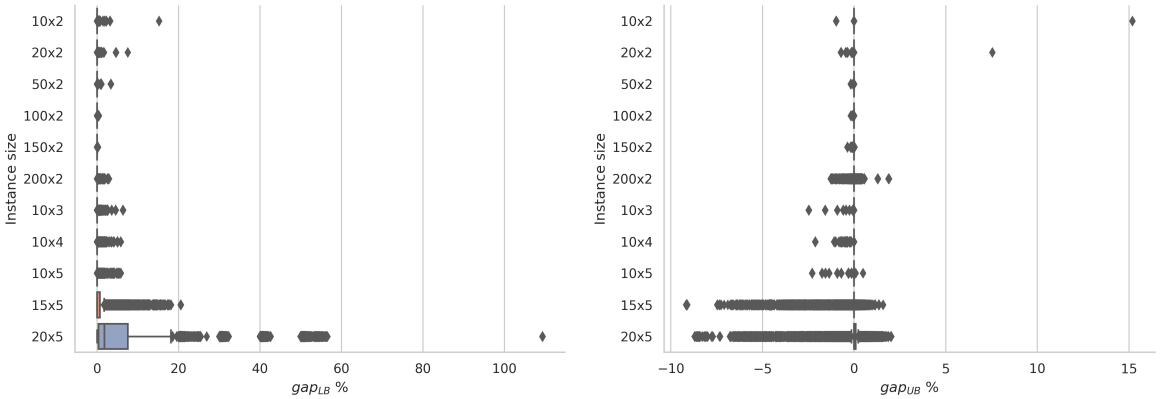

Figure 3.6 – Box plots of GRASP $gap_{LB}\%$ (left plot) and $gap_{UB}\%$ (right plot) values, grouped by instance size, compared to the lower and upper bounds obtained by the C&CG method, respectively. Considers all C&CG solutions, optimal or not.

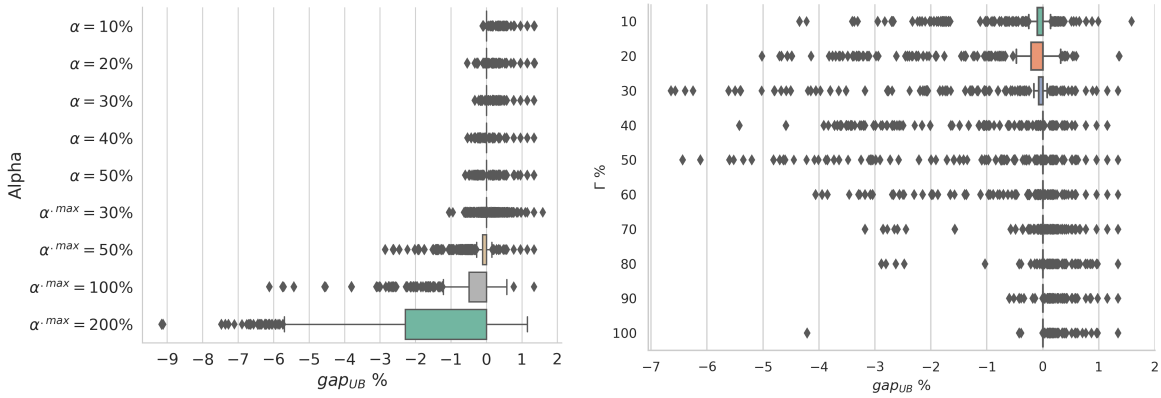

Figure 3.7 – Box plots of $gap_{UB}\%$ values for instance group Ying 15×5, grouped by instance alpha (left) and by $\Gamma\%$ (right), based on all solution values obtained by GRASP and C&CG methods. Considers all C&CG solutions, optimal or not. All GRASP solution values were obtained from 50 independent executions of GRASP.

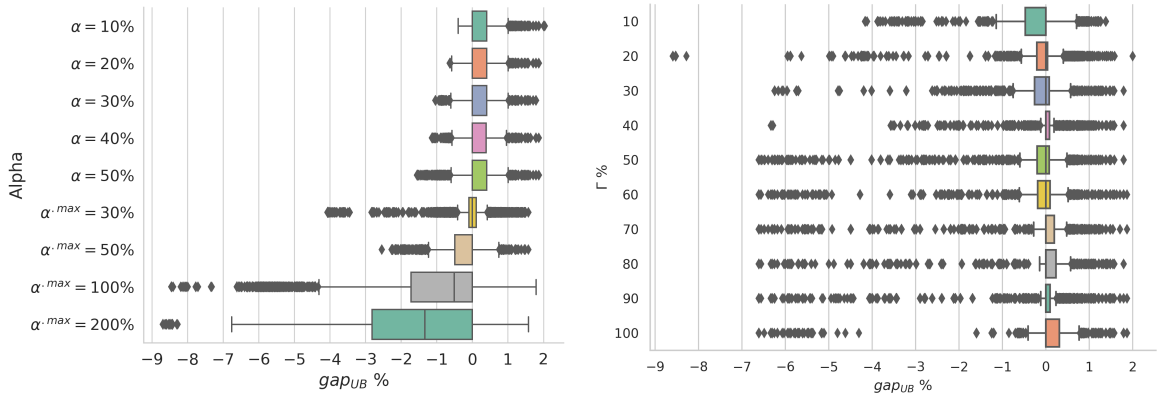

Figure 3.8 – Box plots of $gap_{UB}\%$ values for instance group Taillard 20×5, grouped by instance alpha (left) and by $\Gamma\%$ (right), based on all solution values obtained by GRASP and C&CG methods. Considers all C&CG solutions, optimal or not. All GRASP solution values were obtained from 50 independent executions of GRASP.

3.11 Discussion

This section discusses the results obtained in this chapter, divided into two parts. In the first part, we proposed the first exact solution method for the two-machine robust flow shop problem based on budgeted uncertainty [16]. As a main contribution, we developed a worst-case determination procedure for the problem using polynomial-time dynamic programming. Together with new robust-counterpart formulations, we employed Column-and-Constraint (C&CG) Generation techniques. Extensive experimental results demonstrated that the proposed algorithm effectively obtained optimal robust schedules for small and medium-sized problems (e.g., instances with $m = 2$ and $n \leq 100$). However, it requires more computational power and thus CPU time for larger instances ($m = 2$, $n \geq 150$) and as processing time variability level α increases.

Based on a case study with two representative instances, we have also assessed the trade-off between solution quality and cost, comparing robust solutions to deterministic and stochastic ones. The adoption of the budget of uncertainty avoids the over-conservativeness of conventional robust scheduling approaches and, at the same time, provides a pool of robust schedules, many of which perform well under different levels of realization of uncertainty. Also, according to simulations based on three probability distributions, such robust schedules presented only a small overhead in the expected solution cost.

In the second part of the chapter, we extended our research to the m-machine robust permutation flow shop problem to minimize the worst-case *makespan*. Two solution methods were proposed: an exact algorithm based on C&CG and a greedy randomized adaptive search procedure (GRASP) metaheuristic.

For the sake of performance evaluation, we proposed four sets of benchmark test instances for the robust problem, including instances based on the classical permutation flow shop literature. Af-

ter evaluating both C&CG and GRASP algorithms on this test-bed, GRASP was shown to produce optimal solutions on most small and medium-sized instances (e.g., instances with $n \times m \leq 100$) and associated budget parameters. In fact, it reached a higher proportion of instances solved to optimality than the exact C&CG method itself. Additionally, considering instances not solved to optimality, 75% of the observed GRASP optimality gaps are smaller than 10%.

In a nutshell, the GRASP method was able to process larger problem instances, as high as the Taillard-based 100×50 instances. Extensive experimental results also demonstrated that, compared to the baseline C&CG exact method, the GRASP algorithm was very efficient in obtaining robust schedules, both in solution time and quality (i.e., gaps in solution values).

Chapter 4

Robust scheduling in Oil and Gas exploration

In this chapter, we propose exact solution approaches for the m-machine robust permutation flow shop problem, with the objective of minimizing worst-case total weighted completion time. In the context of the oil and gas industry, this variant of the *flow shop* is associated with the maintenance schedule for oil rigs.

Traditional problem formulations assume that the processing time of each operation does not vary (i.e. the most likely scenario will certainly occur). However, in oil and gas applications like platform maintenance, task duration does fluctuate according to a specific range. Rather than assigning probabilities to different scenarios in order to generate a decision which is optimal in the long run (as in Stochastic Optimization approaches), our goal is to apply Robust Optimization [14] techniques, by considering a set of potentially realizable scenarios and a min-max criteria that returns a solution with the best worst-case performance over the scenario set.

Similarly to the problem studied in the previous chapter, operation processing times are uncertain and vary in a given interval. Moreover, following the concept of budgeted uncertainty, only a subset of operation processing times will deviate to worst-case values. To solve this robust problem, we developed seven robust counterpart formulations, which can be used to derive optimal solutions for medium-sized problem instances by using a Column-and-Constraint Generation algorithm. The efficacy of the solution methods is validated through experiments on three sets of randomly-generated instances. And finally, a case study for maintenance schedule of Brazilian oil platforms is presented.

4.1 Introduction

In the context of the Oil and Gas industry, maintenance scheduling plays a very important role [113, 161, 38]. One major challenge is related to programmed shutdown and maintenance of oil platforms. Due to existing policies in such schedules, a well-defined set of operations must have their order of execution respected. These tasks must be performed on all equipment associated with a specific oil well. For example, they include substituting previously-installed provisional

repairs, corrosion removal, replacing damaged paint, and servicing pipes and water re-injection pumps. Moreover, since maintenance tasks have to be executed on every oil well connected to the platform, they must be closed simultaneously. Oil wells will only reopen for production at the end of the process, as soon as their associated maintenance operations finish.

The aforementioned process can be characterized as a *permutation flow shop scheduling* [109] in which oil-wells are represented by *jobs*, and maintenance tasks by *machines*. Each oil well is closed at the start of the schedule, while its associated equipment undergoes a series of maintenance tasks that always follow the same order. The aim is to find a schedule that minimizes the loss of oil production associated with each oil well's flow rate and for how long it remained closed, i.e., the total weighted completion time (TWCT) objective. Such optimization generates substantial financial gains, as more oil will be produced, in the order of thousands of dollars.

The TWCT criterium is usually associated with production environments where inventory levels and manufacturing cycle times are of critical concern. With a particular interest in minimizing inventory or holding costs, some production environments aim to minimize the total completion time, assuming all jobs are equal in importance. In specific contexts, however, the importance or value of each job may not be the same. For example, jobs may have different unit costs and holding costs. Such costs could be characterized as job weights. As is the case for the oil industry, the cost criterion of each job will depend not only on its completion time but also on its weight, which represents the flow rate of each oil well.

Considering real-world characteristics, solving this scheduling problem is a challenging task and demands efficient solution approaches [54, 112]. Furthermore, job processing times are subject to uncertainty, and no probability distribution is known. A viable alternative, adopted here, consists in applying Robust Optimization to obtain a schedule hedging against worst-case scenarios.

Assuming processing times are uncertain and vary in a given interval, the objective of the present chapter is to provide efficient solutions for the m-machine Robust Permutation Flow Shop with the *total weighted completion time* objective (RPFS-TWCT). The only information required is the lower and upper bounds of processing times, which can be obtained from historical data. We are interested in a job permutation that minimizes the worst-case cost, for any possible realization of job processing times under the budgeted uncertainty set [16]. Unlike other robust optimization models, which provide only one conservative solution, the budgeted approach allows the adjustment of the solution's level of conservatism according to the decision-maker's risk-aversion.

Concerning uncertain processing times, scheduling problems that minimize the total weighted completion time have been studied from various viewpoints, for example, single-machine scheduling heuristics [3], branch-and-bound [108], m-machine heuristics [69], as well as stability analysis methods [127, 79]. To the best of our knowledge, this is the first work to treat the m-machine robust permutation flow shop problem under budgeted uncertainty, which minimizes the *worst-case weighted sum of job completion times*. The solution method is partly based on the column-and-

constraint method designed for the RPFS with *makespan* objective and presented in the previous chapter.

This chapter adopts the following structure. Section 4.2 introduces the classical deterministic Permutation Flow Shop Problem, minimizing total weighted completion time. The m-machine Robust Permutation Flow Shop Problem is presented in Section 4.3, together with seven proposed Robust Counterpart (RC) formulations. Our exact solution approach based on Column-and-Constraint Generation (C&CG) is explained in Section 4.4. An important enhancement to the solution method, which uses a combinatorial branch-and-bound in the master phase of the C&CG method, is discussed in Section 4.5. The experimental results are shown in Section 4.6, based on extensive computational experiments on three sets of randomly-generated problem instances. Section 4.7 brings a case study applied to the oil and gas industry, using real data from the operation history of two Brazilian oil platforms. Finally, Section 4.8 brings the final discussions.

4.2 The deterministic PFSP minimizing total weighted completion time

This section presents the Permutation Flow Shop Problem (PFSP) to minimize the *Total Weighted Completion Time* (TWCT), also known as *total weighted flow time* [109]. For the sake of simplicity, we will refer to this problem as PFSP-TWCT. Following the well-known $\alpha|\beta|\gamma$ [1] notation for scheduling problems, established by [50], this problem is denoted as $F|prmu|\sum w_jC_j$. Since job processing time values are assumed to be known in advance, we will use the term deterministic when referring to this version of the problem. Also, for completeness, we decided to restate the same basic notation and descriptions used in Section 3.2.1, with objective of redefining φ as the TWCT objective function.

The problem can be stated as follows. Consider a production planning process consisting of a set of jobs $\mathbb{J} = [n]$ to be executed in a set of machines $\mathbb{M} = [m]$ [2]. Each job $i \in \mathbb{J}$ has an associated weight w_i and a non-negative processing time $p_{r,i}$ on machine $r \in \mathbb{M}$, forming the matrix $\mathbf{P} \in \mathbb{R}^{+}_{\mathbb{M}\times\mathbb{J}}$. Each job must be processed without preemption on each machine in the same order. At any time, a machine cannot handle more than one job. Also, at any time, a job can only be processed on one machine. We assume intermediate storage between successive machines is unlimited. The permutation flow shop's particularity is that the sequence in which the jobs are to be processed is the same for all machines. Such sequence is defined by a permutation $\sigma : \{1, \ldots, n\} \longrightarrow \mathbb{J}$, with $\sigma(j)$ indicating the jth job to be executed. We call Σ the set of all permutations of n jobs, hence $\sigma \in \Sigma$. Consider an operation $O_{r,\sigma(j)}$, concerning the execution of the jth job on machine r. Its completion

1. Where α represents the machine environment, β stands for job characteristics, and γ symbolizes the objective function. In our case: the flow shop problem, single job permutation, minimizing total weighted completion time.
2. We use the notation $[n] = \{1, \ldots, n\}$.

time, denoted by $C_{r,\sigma(j)}$, can be defined by the recurrence:

$$C_{r,\sigma(j)} = \begin{cases} p_{r,\sigma(j)} & \text{if } r = 1 \text{ and } j = 1, \\ C_{r,\sigma(j-1)} + p_{r,\sigma(j)} & \text{if } r = 1 \text{ and } j > 1, \\ C_{r-1,\sigma(j)} + p_{r,\sigma(j)} & \text{if } r > 1 \text{ and } j = 1, \\ max(C_{r,\sigma(j-1)},\ C_{r-1,\sigma(j)}) + p_{r,\sigma(j)} & \text{if } r > 1 \text{ and } j > 1. \end{cases}$$

The completion time of a job i is defined as its completion time on the last machine $C_{m,i}$. The objective is to find a job sequence that minimizes the total weighed completion times on the final machine, i.e., a permutation σ minimizing $\varphi(\sigma) = \sum_{j\in\{1,\dots n\}} w_{\sigma(j)} C_{m,\sigma(j)}$.

The problem was proved strongly NP-hard by [44] for instances with two or more machines, when all job weights are equal. It was also studied from the viewpoint of probabilistic analysis [70], stability approach [126], heuristics [45, 97, 112, 145], combinatorial branch-and-bound [24], MIP-based branch-and-bound [157, 141] and approximation algorithms [101]. Finally, the problem can be solved with MILP techniques, by adapting the objective function of existing flowshop formulations [135, 136].

4.3 The robust PFSP to minimize the total weighted completion time

Different optimization criteria can be used to search for a robust solution. This study focuses on the *minimax* or *absolute robust* criterion: the robust decision looks for a solution that minimizes the highest objective value over all possible scenarios, following a predefined uncertainty set.

This section starts with a definition of the Robust Permutation Flow Shop - Total Weighted Completion Time (RPFS-TWCT) problem (Section 4.3.1), followed by a description of the underlying budgeted uncertainty set (Section 4.3.2). Then, seven robust counterpart formulations are proposed (Section 4.3.3), based on well-known Mixed-Integer Linear Programming (MILP) formulations for the deterministic problem.

4.3.1 Problem statement

Assume the matrix of individual processing times $\mathbf{P} = \{p_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$ contains uncertain data. A scenario λ is defined as a realization of uncertainty and, for each possible λ, there is a unique matrix of processing times denoted as $\mathbf{P}^\lambda = \{p^\lambda_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$. Let Λ be the set of all possible scenarios λ. Whenever a matrix of processing times $\mathbf{P}^\lambda$ is known, an instance of the deterministic PFSP-TWCT is defined.

Let's redefine $\varphi(\sigma, \mathbf{P}^\lambda)$ as the *total weighted completion time* of a sequence $\sigma \in \Sigma$ given a scenario $\lambda \in \Lambda$. The objective of the RPFS-TWCT is to find a job permutation $\sigma \in \Sigma$ that *minimizes*

the *maximum* possible **total weighted completion time** over all scenarios $\lambda \in \Lambda$:

$$\textbf{RPFS-TWCT:} \quad \min_{\sigma \in \Sigma} \max_{\lambda \in \Lambda} \{\varphi(\sigma, \mathbf{P}^{\lambda})\}. \tag{4.1}$$

For any sequence $\sigma \in \Sigma$, the value

$$\mathcal{Z}(\sigma) := \max_{\lambda \in \Lambda} \{\varphi(\sigma, \mathbf{P}^{\lambda})\} \tag{4.2}$$

is called the *worst-case total weighted completion time* or *robust cost* for σ. A maximizer in (4.2) is called a worst-case scenario for σ.

As mentioned in the introduction, to the best of our knowledge, this is the first research that applies the budgeted uncertainty set on the m-machine robust permutation flow shop problem. Existing works are related to either single or two-machine variants of the problem. In [19], the complexity of the single-machine version of the RPFS-TWCT problem was shown to be weakly NP-hard if $\Gamma = 1$ and strongly NP-hard for $\Gamma > 1$. [159] developed two metaheuristic algorithms to solve the two-machine problem, but with the *makespan* objective.

4.3.2 Budgeted uncertainty set for the RPFS-TWCT problem

In [159], the three classical Robust-Counterpart Optimization (RCO) models [128, 15, 16] are compared in terms of the number of variables, the number of required constraints, and if the respective formulation is linear or not. When compared to the other RCO models [128, 15], the so-called budgeted uncertainty model [16] fits best for robust scheduling problems, by providing a linear formulation that allows adjusting the level of conservatism of the robust solution, without resulting in a substantial increase in problem size. The inclusion of a budget parameter provides a compromise between robustness and optimality. It is possible to adjust the number of coefficients that simultaneously take their largest variations, based on information about the application. For the case of oil-well maintenance, the problem which will be analyzed in the case study section, it is known that the probability of all maintenance tasks simultaneously deviating to their worst-case execution times is low.

Next, the budget uncertainty set for the RPFS-TWCT problem is defined. Bearing in mind that the definition below is equivalent to the one presented in Section 3.7.2, we decided to replicate it here for the sake of completeness. Consider two positive processing time matrices $\overline{\mathbf{P}}$=$\{\overline{p}_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$ and $\widehat{\mathbf{P}}$=$\{\widehat{p}_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$, that represent the nominal value of and the maximum allowed deviation of $\mathbf{P}$, respectively. In order to apply budgeted uncertainty, we introduce the budget parameter $0 \leq \Gamma \leq mn$, which denotes the maximum number of operations whose uncertain processing times can reach their worst-case values. The $budgeted\ uncertainty\ set$ of operation processing times, denoted as $\mathcal{U}^{\Gamma}$, can be defined as follows:

$$\mathcal{U}^{\Gamma} = \left\{ \mathbf{P} = \{p_{r,i}\} \mid p_{r,i} = \overline{p}_{r,i} + \delta_{r,i}\, \widehat{p}_{r,i},\ \delta_{r,i} \in \{0,1\}, \forall r \in \mathbb{M}, \forall i \in \mathbb{J}; \sum_{r=1}^{m} \sum_{i=1}^{n} \delta_{r,i} \leq \Gamma \right\}, \tag{4.3}$$

Given the uncertainty set $\mathcal{U}^{\Gamma}$, each scenario λ is described by one of the infinite matrices in

this set. For a given scenario λ, let $\delta_{r,i}^{\lambda}$ be the value defining the deviation of the processing time regarding the execution of job $i \in \mathbb{J}$ on machine $r \in \mathbb{M}$, i.e., $p_{r,i}^{\lambda} = \overline{p}_{r,i} + \delta_{r,i}^{\lambda} \widehat{p}_{r,i}^{\lambda}$. Therefore, considering all jobs and machines, the total number of operations whose processing time can deviate to its maximum value is limited to Γ. When $\Gamma = 0$, the problem is equivalent to the nominal problem, i.e., the deterministic PFSP-TWCT. If $\Gamma = mn$, we obtain the box uncertainty set [128]. For a given value of Γ, there are $\binom{mn}{\Gamma}$ possible worst-case scenarios, given the budgeted uncertainty set $\mathcal{U}_{\Gamma}$.

4.3.3 Robust counterparts

A number of MILP models were proposed in the literature for the PFSP. We now present the robust counterparts for the PFSP-TWCT, based on the following seven formulations: Wilson [149]; TBA [138]; Wagner-WST2; TS2 and TS3 [136]; Manne [67] and Liao-You [89]. In their original definition, the first five models rely on assignment constraints in order to find the position occupied by each job in the schedule, while the last two apply disjunctive inequalities with Big-M reformulation to determine if a job appears either before or after another job in the sequence. For more details on the rationale behind each deterministic PFSP model, including illustrative diagrams, we refer the reader to [136].

It is worth noting that the first five models were further adapted for the TWCT objective. Such adaptation involved additional constraints based on the Big-M method to appropriately calculate the variables which represent the completion time of each job i on the last machine. These variables, which are not present in the original models, had to be defined for the correct calculation of total weighted completion time.

4.3.3.1 Robust Counterpart for Wilson PFS Model

Wilson [149] proposed a MILP model for the *makespan*-minimizing flow shop scheduling problem, by applying sets of inequality constraints, based on start time variables, of each job on each machine. In this work, we derived a two-stage robust counterpart of his model, for the *total weighted completion time* objective, with the following decision variables:

$Z_{i,j} =$	$\begin{cases} 1, \text{ if } \sigma(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma), \\ 0, \text{ otherwise.} \end{cases}$
$B_{r,j}^{\lambda} =$	start time of job $\sigma(j)$ (in position j) on machine M_r given scenario λ.
$F_i^{\lambda} =$	variable representing the completion time of job i on machine M_m in scenario λ.

Based on the above definitions, variables $Z_{i,j}$ are in the first stage, and variables $B_{r,j}^{\lambda}$ and F_i^{λ} are in the second stage of this robust counterpart. The two-stage robust-counterpart of Wilson model

for the RPFS-TWCT can be formulated as follows:

$$\text{Min } y \tag{4.4}$$

$$\text{st } \sum_{i=1}^{n} w_i F_i^{\lambda} \leq y, \qquad \lambda \in \Lambda, \tag{4.5}$$

$$B_{1,1}^{\lambda} = 0, \qquad \lambda \in \Lambda, \tag{4.6}$$

$$B_{1,j}^{\lambda} + \sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^{\lambda}\right) Z_{i,j} = B_{1,j+1}^{\lambda}, \qquad j = 1,\ldots,n-1, \lambda \in \Lambda, \tag{4.7}$$

$$B_{r,1}^{\lambda} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^{\lambda}\right) Z_{i,1} = B_{r+1,1}^{\lambda}, \qquad r = 1,\ldots,m-1, \lambda \in \Lambda, \tag{4.8}$$

$$B_{r,j}^{\lambda} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^{\lambda}\right) Z_{i,j} \leq B_{r+1,j}^{\lambda}, \; r = 1,\ldots,m-1, j = 2,\ldots,n, \lambda \in \Lambda, \tag{4.9}$$

$$B_{r,j}^{\lambda} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^{\lambda}\right) Z_{i,j} \leq B_{r,j+1}^{\lambda}, \; r = 2,\ldots,m, j = 1,\ldots,n-1, \lambda \in \Lambda, \tag{4.10}$$

$$F_i^{\lambda} \geq B_{m,j}^{\lambda} + \left(\overline{p}_{m,i} + \widehat{p}_{m,i}\delta_{m,i}^{\lambda}\right) Z_{i,j} - Q(1 - Z_{i,j}), i = 1,...,n, j = 1,...,n, \lambda \in \Lambda, \tag{4.11}$$

$$\sum_{i=1}^{n} Z_{i,j} = 1, \qquad j = 1,\ldots,n, \tag{4.12}$$

$$\sum_{j=1}^{n} Z_{i,j} = 1, \qquad i = 1,\ldots,n, \tag{4.13}$$

$$Z_{i,j} \in \{0,1\}, \qquad i,j = 1,\ldots,n, \tag{4.14}$$

$$B_{r,j}^{\lambda} \geq 0, \qquad r = 1,\ldots,m, j = 1,\ldots,n, \lambda \in \Lambda, \tag{4.15}$$

$$F_i^{\lambda} \geq 0, \qquad i = 1,\ldots,n, \lambda \in \Lambda, \tag{4.16}$$

$$y \geq 0. \tag{4.17}$$

The objective function (4.4) and constraint (4.5) state that this formulation aims to find a robust schedule for the processing of n jobs that minimizes the *weighted sum of completion times* of the worst-case scenario, among all possible scenarios $\lambda \in \Lambda$. Constraints (4.6)-(4.10) guarantee that the robust schedule is feasible and that start time variables are appropriately calculated, for each scenario λ. Constraints (4.11) are used to determine the completion time of job i on the last machine m, for each scenario λ. In these constraints, assume Q is a large-enough number. The same assumption is made in all other formulations, using a big-M value Q. Constraints (4.12) and (4.13) are the classical assignment constraints, ensuring, respectively, that each job is assigned to one and only one sequence position, and that each sequence position is filled by one and only one job. Finally, constraints (4.14)-(4.17) define the domain of the variables.

4.3.3.2 Robust Counterpart for TBA PFS Model

Relying on the assignment constraints of Wilson model, Turner and Booth [138] derived a MILP formulation for the PFSP, here called Turner–Booth alternative (TBA) model. After deriving an equivalent mathematical expression for start time variables, in terms of processing and idle times of each job, the authors applied variable substitution techniques, significantly reducing the number of model constraints. We derived a two-stage robust counterpart of this model, for the TWCT objective, with decision variables $Z_{i,j}$ and F_i^{λ} as well as the new ones described below:

$X_{r,j}^{\lambda} =$	idle time on machine M_r before the start of job in sequence position j given scenario λ.

Based on the above definitions, variables $Z_{i,j}$ are in the first stage, and variables $X_{r,j}^{\lambda}$ and F_i^{λ}

are in the second stage of this robust counterpart. The two-stage robust-counterpart of TBA model for the RPFS-TWCT can be formulated as follows:

$$\text{Min } y \tag{4.18}$$

$$\text{st } (4.5), (4.12), (4.13), (4.14), (4.16), (4.17),$$

$$X^{\lambda}_{1,j} = 0, \qquad j = 2, \ldots, n, \lambda \in \Lambda, \tag{4.19}$$

$$\begin{aligned} &\textstyle\sum_{i=1}^{n} \left(\overline{p}_{r-1,i} + \widehat{p}_{r-1,i}\delta^{\lambda}_{r-1,i}\right) Z_{i,1} + \sum_{q=1}^{j-1}\sum_{i=1}^{n} \left(\overline{p}_{r,i} - \overline{p}_{r-1,i}\right) Z_{i,q} \\ &+ \textstyle\sum_{q=1}^{j-1}\sum_{i=1}^{n} \left(\widehat{p}_{r,i}\delta^{\lambda}_{r,i} - \widehat{p}_{r-1,i}\delta^{\lambda}_{r-1,i}\right) Z_{i,q} + \sum_{s=2}^{j} \left(X_{r,s} - X_{r-1,s}\right) \\ &- \textstyle\sum_{i=1}^{n} \left(\overline{p}_{r-1,i} + \widehat{p}_{r-1,i}\delta^{\lambda}_{r-1,i}\right) Z_{i,j} \geq 0, \quad r = 2, \ldots, m, j = 2, \ldots, n, \lambda \in \Lambda, \end{aligned} \tag{4.20}$$

$$\begin{aligned} F^{\lambda}_{i} \geq &\textstyle\sum_{r=1}^{m-1}\sum_{\ell=1}^{n} \left(\overline{p}_{r,\ell} + \widehat{p}_{r,\ell}\delta^{\lambda}_{r,\ell}\right) Z_{\ell,1} + \sum_{q=1}^{j}\sum_{\ell=1}^{n} \left(\overline{p}_{m,\ell} + \widehat{p}_{m,\ell}\delta^{\lambda}_{m,\ell}\right) Z_{\ell,q} \\ &+ \textstyle\sum_{s=1}^{j} X_{m,s} - Q(1 - Z_{i,j}), \qquad i = 1, \ldots, n, j = 1, \ldots, n, \lambda \in \Lambda, \end{aligned} \tag{4.21}$$

$$X^{\lambda}_{r,j} \geq 0, \qquad r = 1, \ldots, m, j = 1, \ldots, n, \lambda \in \Lambda. \tag{4.22}$$

The objective function (4.18) and constraints (4.5), (4.12) and (4.13) are as defined in the previous formulation. Constraints (4.19)-(4.20) guarantee that the robust schedule is feasible and that idle time variables are appropriately calculated, for each scenario λ. Constraints (4.21) are big-M constraints used to determine the completion time of job i on the last machine m, for each scenario λ. For an illustrative diagram, we refer the reader to Figure 2 in [136, p. 1376]. Finally, constraints (4.14), (4.16), (4.17) and (4.22) define the domain of the variables.

4.3.3.3 Robust Counterpart for WST2 PFS Model

Wagner [142] proposed an all-integer programming model for a three-machine deterministic flow shop, later extended to a m-machine MILP model by Stafford [129], and commonly named in the literature as *Wagner model*. In 2002, based on this model and works from other authors, Stafford and Tseng released an improved model called WST and, later on, a second version called WST2 [136], which enforces the initial condition that all jobs are processed on the first machine without any in-sequence machine idleness. In our research, we derived a two-stage robust counterpart of the WST2 model, for the *total weighted completion time* objective, with decision variables $Z_{i,j}$, $X^{\lambda}_{r,j}$ and F^{λ}_{i} as described in the previous formulations, and variables $Y^{\lambda}_{r,j}$:

$Y^{\lambda}_{r,j} =$	idle time of job in sequence position j after it finishes processing on machine M_r given scenario λ.

Variables $Z_{i,j}$ are in the first stage, and variables $X^{\lambda}_{r,j}$, $Y^{\lambda}_{r,j}$ and F^{λ}_{i} are in the second stage of

this robust counterpart. The WST2 model for the RPFS-TWCT can be formulated as follows:

$$\text{Min } y \tag{4.23}$$

$$\text{st } (4.5), (4.12), (4.13), (4.14), (4.16), (4.17), (4.22),$$

$$X^{\lambda}_{1,1} = 0, \qquad \lambda \in \Lambda, \tag{4.24}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,j+1} + X^{\lambda}_{r,j+1} + Y^{\lambda}_{r,j+1} = \sum_{i=1}^{n} \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta^{\lambda}_{r+1,i}\right) Z_{i,j} + X^{\lambda}_{r+1,j+1} + Y^{\lambda}_{r,j}, \qquad r = 2, \ldots, m-1, j = 2, \ldots, n-1, \lambda \in \Lambda, \tag{4.25}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}\right) Z_{i,j+1} + Y^{\lambda}_{1,j+1} = \sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta^{\lambda}_{2,i}\right) Z_{i,j} + X^{\lambda}_{2,j+1} + Y^{\lambda}_{1,j}, \qquad j = 2, \ldots, n-1, \lambda \in \Lambda, \tag{4.26}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,2} + X^{\lambda}_{r,2} + Y^{\lambda}_{r,2} = \sum_{i=1}^{n} \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta^{\lambda}_{r+1,i}\right) Z_{i,1} + X^{\lambda}_{r+1,2}, \qquad r = 2, \ldots, m-1, \lambda \in \Lambda, \tag{4.27}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}\right) Z_{i,2} + Y^{\lambda}_{1,2} = \sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta^{\lambda}_{2,i}\right) Z_{i,1} + X^{\lambda}_{2,2}, \quad \lambda \in \Lambda, \tag{4.28}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,1} + X^{\lambda}_{r,1} = X^{\lambda}_{r+1,1}, \qquad r = 1, \ldots, m-1, \lambda \in \Lambda, \tag{4.29}$$

$$F^{\lambda}_{i} \geq \sum_{p=1}^{j} \sum_{x=1}^{n} \left(\overline{p}_{m,x} + \widehat{p}_{m,x}\delta^{\lambda}_{m,x}\right) Z_{x,p} + \sum_{p=1}^{j} X^{\lambda}_{m,p} - Q(1 - Z_{i,j}), \qquad i = 1, \ldots, n, j = 1, \ldots, n, \lambda \in \Lambda, \tag{4.30}$$

$$Y^{\lambda}_{r,j} \geq 0, \qquad r = 1, \ldots, m, j = 1, \ldots, n, \lambda \in \Lambda. \tag{4.31}$$

The objective function (4.23) and constraints (4.5), (4.12) and (4.13) are as defined in the first formulation. Constraints (4.24)-(4.29) guarantee that the robust schedule is feasible and that idle time variables are appropriately calculated, for each scenario λ. (4.30) are big-M constraints used to determine the completion time of job i on the last machine m, for each scenario λ. Finally, constraints (4.14), (4.16), (4.17), (4.22) and (4.31) define the domain of the variables.

4.3.3.4 Robust Counterpart for TS2 PFS Model

The TS2 MILP model for the regular flow shop is based on an earlier model with the same name that was developed by Tseng et al. [134] for the sequence-dependent setup times flow shop problem. This model uses the job ending or completion time variables employed in other scheduling models, which eliminates the need for the X and Y variables used in Wagner-WST2 model. Besides adapting the TS2 model to the TWCT objective, we also derived a two-stage robust counterpart with variables $Z_{i,j}$ and F^{λ}_{i} as described in the first formulation, along with variables $E^{\lambda}_{r,j}$:

$E^{\lambda}_{r,j} =$	completion time of job in sequence position j after it finishes processing on machine r given scenario λ.

Variables $Z_{i,j}$ are again in the first stage, while variables $E^{\lambda}_{r,j}$ and F^{λ}_{i} are in the second stage of this robust counterpart. The two-stage robust-counterpart of TS2 model for the RPFS-TWCT can

be formulated as:

$$\text{Min } y \tag{4.32}$$

$$\text{st } (4.5), (4.12), (4.13), (4.14), (4.16), (4.17),$$

$$E^{\lambda}_{r,j+1} \geq E^{\lambda}_{r,j} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,j+1},\ r = 2\ldots, m, j = 1, \ldots, n-1, \lambda \in \Lambda, \tag{4.33}$$

$$E^{\lambda}_{r+1,j} \geq E^{\lambda}_{r,j} + \sum_{i=1}^{n} \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta^{\lambda}_{r+1,i}\right) Z_{i,j}, \quad r = 1\ldots, m-1, j = 2, \ldots, n, \lambda \in \Lambda, \tag{4.34}$$

$$E^{\lambda}_{1,j+1} = E^{\lambda}_{1,j} + \sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}\right) Z_{i,j+1} \quad j = 1, \ldots, n-1, \lambda \in \Lambda, \tag{4.35}$$

$$E^{\lambda}_{r+1,1} = E^{\lambda}_{r,1} + \sum_{i=1}^{n} \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta^{\lambda}_{r+1,i}\right) Z_{i,1}, \quad r = 1, \ldots, m-1, \lambda \in \Lambda, \tag{4.36}$$

$$E^{\lambda}_{1,1} = \sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}\right) Z_{i,1} \quad \lambda \in \Lambda, \tag{4.37}$$

$$F^{\lambda}_{i} \geq E^{\lambda}_{m,j} - Q(1 - Z_{i,j}) \quad i = 1, \ldots, n, j = 1, \ldots, n, \lambda \in \Lambda, \tag{4.38}$$

$$E^{\lambda}_{r,j} \geq 0, \quad r = 1, \ldots, m, j = 1, \ldots, n, \lambda \in \Lambda. \tag{4.39}$$

The objective function (4.32) and constraints (4.5), (4.12) and (4.13) are as defined in the first formulation. Constraints (4.33)-(4.37) guarantee that the robust schedule is feasible and that completion time variables are appropriately calculated, for each scenario λ. Constraints (4.38) are big-M constraints used to determine the completion time of job i on the last machine m, for each scenario λ. Finally, constraints (4.14), (4.16), (4.17) and (4.39) define the domain of the variables.

4.3.3.5 Robust Counterpart for TS3 PFS Model

Using an approach similar to the one applied in the TBA model, Tseng and Stafford [136] proposed a MILP formulation for the PFSP called TS3. By applying variable substitution on Wilson model, the start time variable, for a given r and j, is replaced by an expression that combines the sum of the processing times of jobs in sequence positions 1 through $j-1$ on machine 1, and the sum of the processing times of the job in position j on machines 1 through $r-1$, incremented of job's idle times (following each of these same machines). To apply Robust Optimization, we derived a two-stage robust counterpart of the TS3 model, adapted to the *total weighted completion time* objective, with variables $Z_{i,j}$, F^{λ}_{i} and $Y^{\lambda}_{r,j}$ as previously defined.

As in previous formulations, variables $Z_{i,j}$ are in the first stage, while variables $Y^{\lambda}_{r,j}$ and F^{λ}_{i} are

in stage two. The two-stage robust-counterpart of TS3 model can be formulated as follows:

$$\text{Min } y \tag{4.40}$$

$$\text{st } (4.5), (4.12), (4.13), (4.14), (4.16), (4.17), (4.31),$$

$$Y^{\lambda}_{r,1} = 0, \qquad r = 1, \ldots, m-1, \lambda \in \Lambda, \tag{4.41}$$

$$\begin{aligned} &\textstyle\sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i} - \overline{p}_{r,i} - \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,j-1} \\ &+ \textstyle\sum_{q=1}^{r-1} \sum_{i=1}^{n} \left(\overline{p}_{q,i} + \widehat{p}_{q,i}\delta^{\lambda}_{q,i}\right)\left(Z_{i,j} - Z_{i,j-1}\right) \\ &+ \textstyle\sum_{q=1}^{r-1} \left(Y_{q,j} - Y_{q,j-1}\right) \geq 0, \qquad r = 2, \ldots, m, j = 2, \ldots, n, \lambda \in \Lambda, \end{aligned} \tag{4.42}$$

$$\begin{aligned} F^{\lambda}_{i} \geq &\textstyle\sum_{q=1}^{j} \sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}\right) Z_{i,q} + \sum_{r=2}^{m} \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,j} \\ &+ \textstyle\sum_{r=1}^{m-1} Y_{r,j} - Q(1 - Z_{i,j}), \qquad i = 1, \ldots, n, j = 1, \ldots, n, \lambda \in \Lambda. \end{aligned} \tag{4.43}$$

The objective function (4.40) and constraints (4.5), (4.12) and (4.13) are as defined in the first formulation. Constraints (4.41)-(4.42) guarantee that the robust schedule is feasible and that idle time variables are appropriately calculated, for each scenario λ. Constraints (4.43) are big-M constraints used to determine the completion time of job i on the last machine m, for each scenario λ. Finally, constraints (4.14), (4.16), (4.17) and (4.31) define the domain of the variables.

4.3.3.6 Robust Counterpart for Manne PFS Model

Manne [94] proposed a dichotomous-constraints integer programming model for the general job shop problem. The model assures that, for two jobs i and k, only one of each pair of completion-time subtraction constraints can hold, i.e., job i either precedes job k somewhere in the processing sequence, or it does not, thus implying that job k precedes job i. Later, Stafford and Tseng [67] adapted this model to a permutation flow shop (*makespan* objective). Based on this last model, we developed its robust counterpart, adapted for the TWCT objective, with the following decision variables:

$D_{i,k} =$	$\begin{cases} 1, & \text{if job } i \text{ is scheduled any time before job } k \\ 0, & \text{otherwise.} \end{cases}$
$C^{\lambda}_{r,i}$	completion time of job i on machine r given scenario λ.

In this two-stage RO formulation, $D_{i,k}$ are the first-stage variables, while $C^{\lambda}_{r,i}$ are second stage

ones. The robust counterpart for Manne PFS model can be formulated as follows.

$$\text{Min } y \tag{4.44}$$

$$\text{st } \sum_{i=1}^{n} w_i C^{\lambda}_{m,i} \leq y, \qquad \lambda \in \Lambda, \tag{4.45}$$

$$C^{\lambda}_{1,i} \geq \overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}, \qquad i = 1,\ldots,n, \lambda \in \Lambda, \tag{4.46}$$

$$C^{\lambda}_{r,i} - C^{\lambda}_{r-1,i} \geq \overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}, \qquad r = 2,\ldots,m, i = 1,\ldots,n, \lambda \in \Lambda, \tag{4.47}$$

$$C^{\lambda}_{r,i} - C^{\lambda}_{r,k} + QD_{i,k} \geq \overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}, \qquad r = 1,\ldots,m,$$
$$i = 1,\ldots,n-1, k = i+1,\ldots,n, \lambda \in \Lambda, \tag{4.48}$$

$$C^{\lambda}_{r,i} - C^{\lambda}_{r,k} \leq Q(1 - D_{i,k}) - (\overline{p}_{r,k} + \widehat{p}_{r,k}\delta^{\lambda}_{r,k}), \qquad r = 1,\ldots,m,$$
$$i = 1,\ldots,n-1, k = i+1,\ldots,n, \lambda \in \Lambda, \tag{4.49}$$

$$C^{\lambda}_{r,i} \geq 0, \qquad r = 1,\ldots,m; i = 1,\ldots,n, \lambda \in \Lambda, \tag{4.50}$$

$$D_{i,k} \in \{0,1\}, \qquad i = 1,\ldots,n-1, k = i+1,\ldots,n, \tag{4.51}$$

$$y \geq 0. \tag{4.52}$$

The objective function (4.44) and constraints (4.45) represent the worst-case total weighted completion time objective, i.e., the minimization of the maximum sum of the weighted completion time of all jobs on the last machine, given all scenarios $\lambda \in \Lambda$. Constraints (4.46) insure that the completion time of each job on machine 1 occurs no earlier than the duration of that job's processing time on machine 1. Constraints (4.47) insure that each job's completion time on machine r is no earlier than the job's completion time on machine $r-1$ plus the job's processing time on machine r (with or without deviation). Constraints (4.48) and (4.49) are the paired disjunctive constraints, which insure that job i either precedes job k or follows job k in the sequence, but not both. Finally, constraints (4.50)-(4.52) define the domain of the variables.

4.3.3.7 Robust Counterpart for Liao-You PFS Model

Liao and You [89] made algebraic combinations of each pair of Manne disjunctive inequality constraints. As a result, they obtained one equality constraint associated to a surplus variable, $q_{r,i,k}$, related to the precedence relationship of jobs i and k on machine r. To ensure feasibility, a second constraint was added to impose an upper bound on these surplus variables. Based on Liao-You model (makespan objective), we developed its robust counterpart, adapted for the *total weighted completion time* objective, with the following additional decision variables:

$S^{\lambda}_{r,i}$	start time of job i on machine r given scenario λ.
$q^{\lambda}_{r,i,k}$	surplus variable related to the precedence relationship of jobs i and k on machine r given scenario λ.

In this two-stage RO formulation, $D_{i,k}$ are, as in the previous model, the first-stage variables, while $S^{\lambda}_{r,i}$ and $q^{\lambda}_{r,i,k}$ are on the second stage. The robust counterpart for Liao-You PFS model can be

formulated as follows.

$$\text{Min } y \tag{4.53}$$

$$\text{st } (4.51), (4.52),$$

$$\sum_{i=1}^{n} w_i(S^{\lambda}_{m,i} + \overline{p}_{m,i} + \widehat{p}_{m,i}\delta^{\lambda}_{m,i}) \leq y, \qquad \lambda \in \Lambda, \tag{4.54}$$

$$S^{\lambda}_{r,i} + \overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i} \leq S^{\lambda}_{r+1,i}, \qquad r = 1,\ldots,m-1, i = 1,\ldots,n, \lambda \in \Lambda, \tag{4.55}$$

$$S^{\lambda}_{r,i} - S^{\lambda}_{r,k} + QD_{i,k} - (\overline{p}_{r,k} + \widehat{p}_{r,k}\delta^{\lambda}_{r,k}) = q^{\lambda}_{r,i,k}, \qquad r = 1,\ldots,m,$$
$$i = 1,\ldots,n-1, k = i+1,\ldots,n, \lambda \in \Lambda, \tag{4.56}$$

$$Q - (\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}) - (\overline{p}_{r,k} + \widehat{p}_{r,k}\delta^{\lambda}_{r,k}) \geq q^{\lambda}_{r,i,k}, \qquad r = 1,\ldots,m,$$
$$i = 1,\ldots,n-1, k = i+1,\ldots,n, \lambda \in \Lambda, \tag{4.57}$$

$$S^{\lambda}_{r,i} \geq 0, \qquad r = 1,\ldots,m; i = 1,\ldots,n, \lambda \in \Lambda, \tag{4.58}$$

$$q^{\lambda}_{r,i,k} \geq 0, \qquad r = 1,\ldots,m; i = 1,\ldots,n-1, k = i+1,\ldots,n, \lambda \in \Lambda. \tag{4.59}$$

The objective function (4.53) and constraints (4.54) represent the worst-case total weighted completion time objective, i.e., the minimization of the maximum sum of the weighted completion time of all jobs on the last machine, given all scenarios $\lambda \in \Lambda$. Constraints (4.55) insure that each job's start time on machine $r+1$ is no earlier than the job's start time on machine r plus the job's processing time on machine r (with or without deviation). Constraints (4.56) and (4.57) are the paired disjunctive constraints, which insure that job i either precedes job k or follows job k in the sequence, but not both. Finally, constraints (4.51), (4.52), (4.58) and (4.59) define the domain of the variables.

RC Model	Binary variables	Continuous variables	Constraints
Wilson TBA WST2 TS2 TS3	$\mathcal{O}(n^2)$	$\mathcal{O}(\lambda mn)$	$\mathcal{O}(\lambda(n^2+mn))$
Manne	$\mathcal{O}(n^2)$	$\mathcal{O}(\lambda mn)$	$\mathcal{O}(\lambda mn^2)$
Liao-You	$\mathcal{O}(n^2)$	$\mathcal{O}(\lambda mn^2)$	$\mathcal{O}(\lambda mn^2)$

Table 4.1 – Size complexity of the RPFS-TWCT robust-counterpart MILP models.

Table 4.1 presents the size complexity of each robust counterpart MILP model presented in this section. The number of binary variables remains the same of the original deterministic models, as they consist of first-stage variables. Also observe that the number of continuous variables as well as the number of constraints grow proportionally to the number of scenarios λ, as expected in robust counterpart formulations. Finally, the number of constraints in the assignment-based models (first five models in Table 4.1) is now quadratic in n, due to the calculation of the weighted completion time of each job, which requires the use of n^2 big-M constraints.

Solving each of the aforementioned models, for all possible scenarios $\lambda \in \Lambda$, is unrealistic. Therefore, in the next section, we will describe an algorithm capable of obtaining optimal results for

RPFS-TWCT by considering a subset of relevant scenarios.

4.4 Column-and-Constraint Generation applied to the RPFS-TWCT

This section presents an exact method for solving RPFS-TWCT under budgeted uncertainty. Our approach is based on a cutting plane procedure for two-stage RO problems, called Column-and-Constraint Generation (C&CG), recently applied in the efficient solution of robust scheduling problems [117, 124, 82], as well as in the exact solution methods proposed in the previous chapter. Besides generating new constraints, as usual in this kind of method, each cutting plane of C&CG is also associated with a set of new decision variables for the recourse problem [162].

Given one of the robust counterparts presented in Section 4.3, the main idea is to relax it into a master problem (MP) where each robust constraint is written only for a finite subset of the uncertainty set, i.e., for a $\Theta \subseteq \Lambda$. Then, given a feasible solution to the MP, this solution is checked for feasibility over the whole set Λ, by solving a separation subproblem (SP). If the SP solution indicates that one or more robust constraints become infeasible, the uncertainty set is expanded by adding one or more scenario vectors to Θ. Whenever the master problem is augmented, according to the column-and-constraint generation procedure, the process is repeated.

For the RPFS-TWCT problem, the MP solution represents a permutation σ where $\sigma(j)$ is the order in which job j is executed. The separation problem is then solved by the worst-case procedure, which, given the sequence σ, returns the highest possible *total weighted completion time* under uncertainty set $\mathcal{U}^{\Gamma}$. Since the uncertainty set $\mathcal{U}^{\Gamma}$, defined in Section 4.3, is polyhedral, the number of possible extreme solutions that can be fetched by the procedure is finite, and the C&CG algorithm certainly terminates [162].

4.4.1 C&CG algorithm

We describe the solution method in a general way that can be applied to any two-stage RO formulation from Section 4.3.3. Following the structure of the C&CG method, we define the Master Problem (MP) by choosing an appropriate 2-stage RO formulation. Considering $\Theta = \{\lambda_1, \ldots, \lambda_v\} \subseteq \Lambda$ a subset of scenarios, let $\mathcal{F}_{model}$ and $\mathcal{R}_{model}$ be the set of corresponding first-stage and recourse decision variables of the model, respectively. For instance, $\mathcal{F}_{Wilson} = \{\mathrm{Z_{i,j}}, \forall i, j = 1, \ldots, n\}$ and $\mathcal{R}_{Wilson} = \{\mathrm{B}^{(\lambda)}_{r,j}, \mathrm{F}^{(\lambda)}_{r,j}, \forall \lambda \in \Theta, r = 1, \ldots, m, j = 1, \ldots, n\}$. The master problem (MP) is solved iteratively, with each step generating a subset of problem constraints and associated recourse variables $\mathcal{R}$, regarding one newly-generated scenario $\lambda_v \in \Theta$. The subset of scenarios Θ is iteratively enlarged by solving the associated subproblem at each iteration.

In order to generate the scenarios defined by Θ, we assume that an oracle can obtain an optimal

solution to the worst-case subproblem, based on the current MP solution. At iteration v, for a given value of MP first-stage decision variables $\mathcal{F}$, the subproblem SP is defined as:

$$\text{(SP)} \quad \mathcal{Z}(\sigma) = \max_{\lambda_v \in \Lambda} \varphi(\sigma, \mathbf{P}^{\lambda_v}) \tag{4.60}$$

where job permutation σ is derived using MP optimal values of first-stage variables $\mathcal{F}$ at iteration v (either $Z_{i,j}^{(v)}$ or $D_{i,j}^{(v)}$, depending on the model). The oracle used to find the optimal solution $\lambda_{(v)}^*$ for (SP) is the worst-case MILP described in Section 4.4.2.

The C&CG method is presented in Algorithm 6, where LB denotes the lower bound, UB denotes the upper bound, v is the iteration counter, and Θ is the set of worst-case scenarios generated by the method. The procedure starts by considering Θ with a single scenario in which no operation presents processing time oscillation, and stops whenever the tolerance of optimality $\epsilon \in \mathbb{R}^+$ is reached. It returns the optimal solution value of the robust problem, along with the first-stage variables $\mathcal{F}^*$, which represent the optimal permutation σ^*.

4.4.2 Worst-case evaluation based on a MILP model

Solving the SP problem in (4.60) consists in determining the worst-case realization under the budgeted uncertainty set $\mathcal{U}^\Gamma$, for a specific sequence of jobs $\sigma = \{\sigma(j), j = 1, \ldots, n\}$. From equation (4.2), given a protection level Γ and a schedule σ, we extend the definition of *worst-case total weighted completion time* or *robust cost* to $\mathcal{Z}(\sigma, \Gamma)$ with the equation:

$$\mathcal{Z}(\sigma, \Gamma) := \max_{\mathbf{P}^\lambda \in \mathcal{U}^\Gamma} \{\varphi(\sigma, \mathbf{P}^\lambda)\}. \tag{4.61}$$

We assume that parameter Γ, from the budgeted uncertainty set, is a non-negative integer. Since any worst-case realization will use as much budget of uncertainty as possible, we can expect that, for the optimal solution of (4.61), with worst-case scenario $\lambda_{(v)}^*$, $\sum_{r=1}^{m} \sum_{i=1}^{n} \delta_{r,i}^{\lambda^*} = \Gamma$.

The worst-case scenario, and associated robust cost, can be obtained by solving the following proposed SP MILP. As input parameters, besides the processing time matrices $\overline{p}_{r,i}$ and $\widehat{p}_{r,i}$, the SP MILP requires the budget parameter Γ along with the sequence of jobs σ^*, provided by the current Master Problem solution $\mathcal{F}^*$. Since the proposed SP MILP relies on an assignment-based formulation, an equivalent input matrix of assignment values $z_{i,j}^*$ needs to be derived in the following way:

$$z_{i,j}^* = \begin{cases} 1, \text{ if } \sigma^*(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma^*) \\ 0, \text{ otherwise.} \end{cases} .$$

Algorithm 6: Column-and-constraint generation algorithm for the RPFS-TWCT problem

Set $LB = -\infty, UB = +\infty, v = 1, \Theta = \{\lambda_{(0)} : \delta_{r,i}^{(0)} = 0, \forall r = 1, \ldots, m, i = 1, \ldots, n\}$
while (UB − LB)/LB $> \epsilon$ **do**
  if *model*=Wilson **then** Solve the MP defined in Section 4.3.3.1 with $\Lambda := \Theta$
  if *model*=TBA **then** Solve the MP defined in Section 4.3.3.2 with $\Lambda := \Theta$
  if *model*=WST2 **then** Solve the MP defined in Section 4.3.3.3 with $\Lambda := \Theta$
  if *model*=TS2 **then** Solve the MP defined in Section 4.3.3.4 with $\Lambda := \Theta$
  if *model*=TS3 **then** Solve the MP defined in Section 4.3.3.5 with $\Lambda := \Theta$
  if *model*=Manne **then** Solve the MP defined in Section 4.3.3.6 with $\Lambda := \Theta$
  if *model*=Liao-You **then** Solve the MP defined in Section 4.3.3.7 with $\Lambda := \Theta$
  Let $(\mathcal{F}^*_{(v)},\ y^*,\ \mathcal{R}^*_{model})$ be the MP optimal solution
  Update LB := max $\left[\text{LB},\ y^*\right]$
  Call the oracle to solve subproblem (SP) in (4.60) with $\mathcal{F} := \mathcal{F}^*_{(v)}$
  Let $(\mathcal{Z}^*_{(v)},\ \lambda^*_{(v)})$ be the SP optimal solution value and associated worst-case scenario, respectively
  Update UB := min $\left[\text{UB},\ \mathcal{Z}^*_{(v)}\right]$
  if (UB − LB)/LB $> \epsilon$ **then**
    Create recourse decision variables $\mathcal{R}_{(v)}$ for scenario $\lambda^*_{(v)}$ on MP
    Update $\mathcal{R}_{model} := \mathcal{R}_{model} \cup \{\mathcal{R}_{(v)}\}$
    if *model*=Wilson **then** Generate MP constraints (4.5)-(4.11),(4.15)&(4.16) for $\lambda^*_{(v)}$
    if *model*=TBA **then** Generate MP constraints (4.5),(4.16),(4.19)-(4.22) for $\lambda^*_{(v)}$
    if *model*=WST2 **then** Generate MP constraints (4.5),(4.16),(4.22),(4.24)-(4.31) for $\lambda^*_{(v)}$
    if *model*=TS2 **then** Generate MP constraints (4.5),(4.16),(4.33)-(4.39) for $\lambda^*_{(v)}$
    if *model*=TS3 **then** Generate MP constraints (4.5),(4.16),(4.31),(4.41)-(4.43) for $\lambda^*_{(v)}$
    if *model*=Manne **then** Generate MP constraints (4.45)-(4.50) for $\lambda^*_{(v)}$
    if *model*=Liao-You **then** Generate MP constraints (4.54)-(4.59) for $\lambda^*_{(v)}$
    Update $\Theta := \Theta \cup \{\lambda^*_{(v)}\}$ and set $(v) := (v + 1)$
  end
end
Return UB, $\mathcal{F}^*_{(v)}$

Worst-case MILP - Problem variables	
$\Delta_{r,i} =$	$\begin{cases} 1, & \text{if job } i \text{ will have its processing time deviated on machine } r \\ 0, & \text{otherwise.} \end{cases}$
$C_{r,j} =$	completion time of job $\sigma(j)$ (in position j) on machine r.
$E_{r,j} =$	auxiliary variable, equals to $\min(C_{r,j-1}, C_{r-1,j})$.
$A_{r,j} =$	auxiliary variable, equals to $\mid C_{r,j-1} - C_{r-1,j} \mid$.
$D_{r,j} =$	auxiliary variable, models the disjunction used to calculate $A_{r,j}$.

The worst-case MILP is stated as follows:

$$\text{Max } \sum_{i=1}^{n} w_i \sum_{j=1}^{n} C_{m,j} z^*_{i,j} \tag{4.62}$$

$$\text{st } C_{1,1} = \sum_{i=1}^{n} (\overline{p}_{1,i} + \widehat{p}_{1,i}\Delta_{1,i}) z^*_{i,1}, \tag{4.63}$$

$$C_{1,j} = C_{1,j-1} + \sum_{i=1}^{n} (\overline{p}_{1,i} + \widehat{p}_{1,i}\Delta_{1,i}) z^*_{i,j}, \quad j = 2,\ldots,n, \tag{4.64}$$

$$C_{r,1} = C_{r-1,1} + \sum_{i=1}^{n} (\overline{p}_{r,i} + \widehat{p}_{r,i}\Delta_{r,i}) z^*_{i,1}, \quad r = 2,\ldots,m, \tag{4.65}$$

$$E_{r,j} \leq C_{r,j-1}, \quad j = 2,\ldots,n, r = 2,\ldots,m, \tag{4.66}$$

$$E_{r,j} \leq C_{r-1,j}, \quad j = 2,\ldots,n, r = 2,\ldots,m, \tag{4.67}$$

$$A_{r,j} \geq C_{r,j-1} - C_{r-1,j}, \quad j = 2,\ldots,n, r = 2,\ldots,m, \tag{4.68}$$

$$A_{r,j} \geq -(C_{r,j-1} - C_{r-1,j}), \quad j = 2,\ldots,n, r = 2,\ldots,m, \tag{4.69}$$

$$A_{r,j} \leq C_{r,j-1} - C_{r-1,j} + QD_{r,j}, \quad r = 2,\ldots,m, j = 2,\ldots,n, \tag{4.70}$$

$$A_{r,j} \leq -(C_{r,j-1} - C_{r-1,j}) + Q(1 - D_{r,j}), \quad r = 2,\ldots,m, j = 2,\ldots,n, \tag{4.71}$$

$$C_{r,j} \leq \sum_{i=1}^{n} (\overline{p}_{r,i} + \widehat{p}_{r,i}\Delta_{r,i}) z^*_{i,j} + E_{r,j} + A_{r,j}, \quad r = 2,\ldots,m, j = 2,\ldots,n, \tag{4.72}$$

$$\sum_{r=1}^{m} \sum_{i=1}^{n} \Delta_{r,i} \leq \Gamma\,, \tag{4.73}$$

$$\Delta_{r,i} \in \{0,1\}, \qquad r = 1,\ldots,m, i = 1,\ldots,n, \tag{4.74}$$

$$C_{r,j} \geq 0, \qquad r = 1,\ldots,m, j = 1,\ldots,n, \tag{4.75}$$

$$A_{r,j} \geq 0,\ E_{r,j} \geq 0,\ D_{r,j} \in \{0,1\},\ r = 1,\ldots,m, j = 1,\ldots,n, \tag{4.76}$$

The objective function (4.62) states that, given a fixed job permutation $z^*_{i,j}$, this formulation aims to find a worst-case processing time scenario that maximizes the *weighted sum of completion times*, among all possible scenarios defined by $\mathcal{U}^{\Gamma}$. Constraints (4.63)-(4.64) are used to determine the completion time of the jobs on the first machine, while constraints (4.65) define the completion time of the first job on each machine r. For each machine r and job position j, constraints (4.66) and (4.67) are used to calculate the minimum value between the completion time of the previous job on the same machine ($C_{r,j-1}$) and the completion time of the same job on the previous machine ($C_{r-1,j}$). Constraints (4.68)-(4.69), together with disjunctive constraints (4.70)-(4.71) are used to determine the absolute value of the difference between $C_{r,j-1}$ and $C_{r-1,j}$. These absolute values are used to define the completion time $C_{r,j}$. Constraints (4.72) ensure that the completion time $C_{r,j}$ is bounded by the processing time of job $\sigma^*(j)$ (in position j) on machine r, plus the maximum of $C_{r,j-1}$ and $C_{r-1,j}$, which is equivalent to the minimum of these two variables ($E_{r,j}$) plus the absolute difference between the same two variables ($A_{r,j}$). Constraints (4.73) define the budget of uncertainty regarding the maximum allowed processing time deviations $\sum_{r=1}^{m} \sum_{i=1}^{n} \Delta_{r,i}$ given the execution of all jobs i on all machines r. Finally, constraints (4.74)-(4.76) define the domain of the variables.

We employed two strategies to improve the performance of the SP MILP model. First, we adopted a problem-specific method when calculating Big-M values, where each Q value varies according to the constraint it belongs to. Second, in order to strengthen the formulation, the following valid inequality was added, improving solution times by a factor of 10:

$$A_{r,j} = C_{r,j-1} + C_{r-1,j} - 2 \times E_{r,j}. \tag{4.77}$$

The optimal solution of this MILP model, represented by $\Delta^*_{r,i}$ values, consists in a valid worst-

case scenario λ^* under budget uncertainty set $\mathcal{U}^{\Gamma}$. Remark that $\Delta^*_{r,i}$ values are used to define $\delta^{\lambda}_{r,i}$ values for the scenario added to set Θ, and used in the 2-stage RO models.

In the experiments shown in Section 4.6, the convergence of the C&CG method was also accelerated by generating multiple worst-case scenarios at each iteration, whenever possible.

Our computational experiments have evidenced that the limits of the proposed C&CG solution method lie in the solution of the Master Problem. In particular, we observed a high proportion of time spent when solving Master Problems for instances with 15 jobs and 5 machines. Nonetheless, for the oil and gas maintenance problem at hand, improved solutions are needed for instances of this size. Therefore, in the next section, we will propose an algorithm enhancement to overcome this limitation.

4.5 Hybrid C&CG Method

Unsurprisingly, the C&CG will suffer of computational limitation as instance size grows, in particular when solving the Master Problem. For this reason, we devise an improved MP solution method, which brings a combinatorial branch-and-bound inside the MILP solver tree structure.

With this new approach, we implemented an alternative Master Problem solution method for assignment-based and dichotomous-based Robust Counterparts. Similarly to the method presented in Algorithm 6, the alternative MP solution method relies on a RC model invoked in an iterative way, based on a list of *C&CG cuts* provided. We denote as *Hybrid C&CG Method* the C&CG solution method that incorporates this new MP solution technique. The main advantage relies on the alternative branching strategy employed, which provides new information used to prune nodes, as well as a powerful combinatorial lower bound.

The implementation of the hybrid C&CG method was based on the CPLEX solver 20.1. Based on its branch callback, we developed a combinatorial branch-and-bound emulation similar to [114], which will be described next.

4.5.1 Branching strategy

Consider the search tree of the classic flow shop combinatorial branch-and-bound [76], depicted in Figure 4.1(a). The root node (at level 0) represents the *null schedule*. A given node N at level s represents a *partial schedule* $\sigma = (\sigma(1), \ldots, \sigma(s))$ of size s, indicating that job $\sigma(j)$ occupies the j-th position on each machine, for $1 \leq j \leq s$, where $1 \leq s \leq n$. By placing any unscheduled job i in position $(s+1)$, we produce a child node $\sigma i = (\sigma(1), \ldots, \sigma(s), i)$, in level $s+1$.

It is clear that the flow shop search tree requires several branches at each node. However, CPLEX allows the creation of at most two branches at a node. To circumvent this limitation and produce more than two branches, we must emulate multi-way branching by binary branching. To

accomplish that, instead of generating the branching tree of Figure 4.1(a), we create a branching structure following the diagram in Figure 4.1(b). Consider an arbitrary node N from Figure 4.1(a). The new branching scheme produces exactly the same offsprings of each original node, but in multiple levels. In this case, one branch is always one of the children to be created (here called a *permutation branch*), while the second branch is a duplicate of the parent node N, which we call a *meta node*.

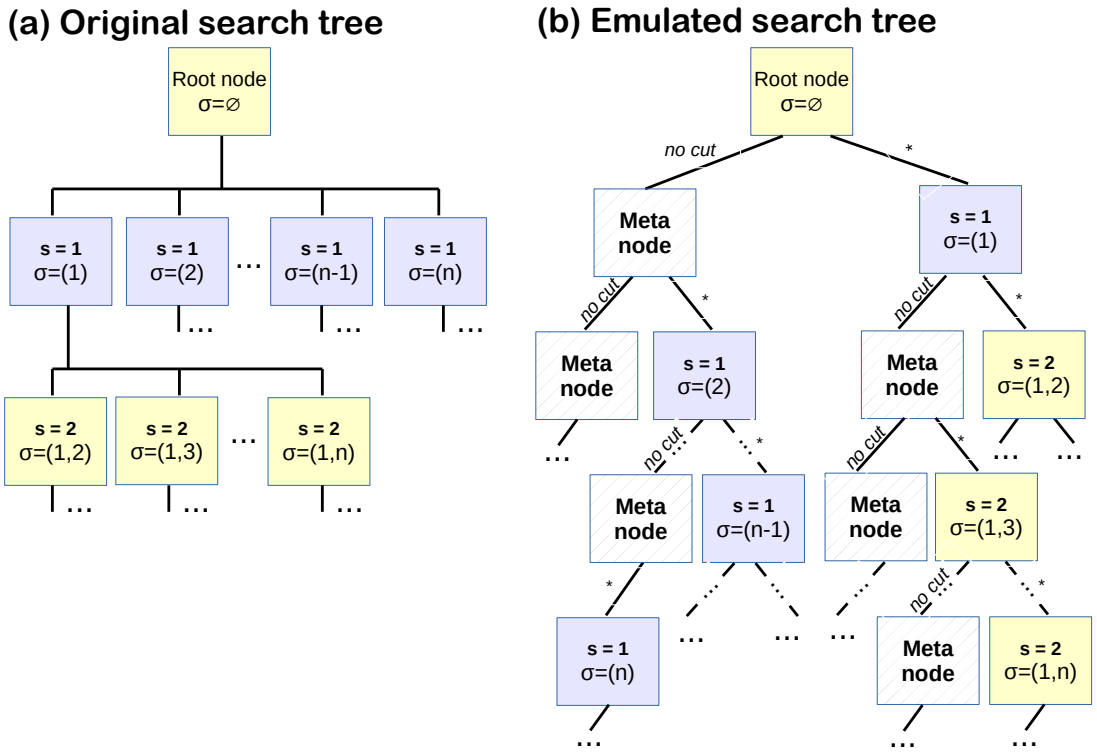

Figure 4.1 – (a) Search tree of the deterministic flow shop combinatorial branch-and-bound. (b) Diagram illustrating how flow shop multi-way branching was performed in CPLEX.

Whenever a new *permutation branch* is created, an unscheduled job i will be fixed in position j of the partial permutation σ. This new partial permutation has to be reflected, in some way, on the node information manipulated by CPLEX, via a set of *node cuts*. For the flow shop MILP models at hand, this means one or more binary variables must have their bounds fixed. Observe that the other branch created, which contains the *meta node*, will receive no additional *node cuts* associated to it, but will receive additional information about the partial sequence generation.

For assignment-based flow shop models, variables Z will be fixed:

• Job i occupies sequence position j:

$$Z_{i,j} := 1; \tag{4.78}$$

• Job i cannot occupy any other position k rather than position j:

$$Z_{i,k} := 0, \forall k \neq j; \tag{4.79}$$

• No other job $\ell \neq i$ can occupy position j:

$$Z_{\ell,j} := 0, \forall \ell \neq i. \tag{4.80}$$

For dichotomous-based flow shop models, partial order variables D will be fixed:

• Set all jobs ℓ that come before job i in partial permutation σ:

$$D_{\ell,i} := 1, D_{i,\ell} := 0, \forall \ell \in \sigma, \tag{4.81}$$

• All jobs ℓ that have not been scheduled yet will necessarily come after job i:

$$D_{i,\ell} := 1, D_{\ell,i} := 0, \forall \ell \in U. \tag{4.82}$$

4.5.2 Improved lower bound

When solving the Master Problem, at each node of the B&B tree, in addition to the branching strategy above, an extended combinatorial lower bound can be applied as an additional criterion to prune nodes. Consider the MP is being solved at iteration v of the *hybrid C&CG method*. At this point, a set of $v-1$ *C&CG cuts* (i.e., violated scenarios λ) has already been generated and applied to the MP model, as explained in Section 4.4.1. The list of existing *C&CG cuts* can be then used to calculate the following combinatorial lower bound LB_{MP}:

$$LB_{MP} = \max_{\lambda \in \Lambda} LB_{det}(P^{\lambda}), \tag{4.83}$$

where LB_{det} is the lower bound of the deterministic PFSP-TWCT, assuming scenario λ and processing time matrix P^{λ}.

To calculate (4.83), we applied the tight lower bound for the deterministic problem described by Chung et al. [24]. These authors developed a branch and bound algorithm to solve the m-machine permutation flowshop problem, which assumes that a partial permutation is defined at each step. In their work, they considered two possible objectives: the unweighted and weighted total flow-time (i.e., TWCT). Their solution method efficiently handles test problems with $n \leq 15$, thanks to an improved machine-based lower bound, together with a dominance test for pruning nodes.

It is worth noting that, despite the overhead from the combinatorial branch-and-bound emulation, the use of the lower bound LB_{MP} to prune nodes has proved to be essential to the performance gains obtained with this new Master Problem solution method.

4.6 Experimental results

We conducted extensive experiments to assess the performance of the C&CG solution method as well as the proposed Robust Counterpart formulations.

4.6.1 Test instances

In the flow shop literature, there is no set of benchmark instances for the *total weighted completion time* objective. In order to verify the effectiveness of the proposed algorithms, our experiments were based on three instance sets[3] obtained by adapting a robust PFSP instance generator described by Ying [159].

(i) **Two-machine robust PFSP instances with 10 jobs (10x2)**. In his work, Ying [159] proposed six groups of instances, each one with a different number of jobs $n \in \{10, 20, 50, 100, 150, 200\}$. The expected processing time $\bar{p}_{r,i}$ ($r = 1, 2$; $i = 1, \ldots, n$) is an integer drawn from the uniform distribution [10, 50] and the largest processing time deviation is set as a fixed ratio of the expected processing time (i.e., $\widehat{p}_{r,i} = \alpha \bar{p}_{r,i}$), with $\alpha \in \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Ten instances were generated for each combination of n and α, for a total of 300 test instances.

(ii) **Robust PFSP instances with 3, 4 and 5 machines**. Following the instance generation algorithm of Ying [159], we generated random instances with sizes $n \times m \in \{10 \times 3, 10 \times 4, 10 \times 5, 15 \times 5\}$. Ten instances were generated for each combination of $m \times n$ and α, for a total of 200 test instances.

(iii) **Robust PFSP instances with random processing time deviations**. For each instance of the previous two sets with variability level $\alpha = 10\%$, we generated 4 new instances with distinct variability levels $\alpha_{r,i}$ for each operation $O_{r,i}$. First, we define a maximum variability level $\alpha^{max} \in \{30\%, 50\%, 100\%, 200\%\}$. Then, in each generated instance, the variability level $\alpha_{r,i}$ of each operation $O_{r,i}$ is drawn from a uniform distribution in the interval $[0, \alpha^{max})$. Therefore, the maximum processing time deviation of each operation equals $\widehat{p}_{r,i} = \alpha_{r,i} \bar{p}_{r,i}$. The idea behind this new set is to generate instances whose operation processing time deviation follow a completely random behavior, when compared to the previous sets. This way, we will be able to assess the impacts of such behavior on the solution method.

Since all instances above are related to the *makespan* objective, no job weight information is available. Thus, for each instance, job weights ($w_j, \ \forall j \in \mathbb{J}$) were randomly generated, according to a uniform distribution in the interval $[1, 100]$. These values are based on the job weight distribution from the real-world instances under study.

4.6.2 Computational environment and model observations

The C&CG algorithm was coded in Julia 1.6.0. CPLEX solver 20.1 was used to solve the Master Problems (MP) and Gurobi solver 9.1 was used to solve the subproblems SP, since it obtained improved performance in preliminary experiments. The MILP time limit was set to 7,200 s and the number of threads was set to 16. All experiments were conducted on a workstation with an Intel Xeon® CPU E5640 @2.67GHz with 32 GB RAM, under Ubuntu 18.04 LTS. In the C&CG algorithm,

3. All test instances are available at https://github.com/levorato/RPFS_Budget_TWCT.

optimality gap tolerance ϵ was set to 10^{-8}.

In the literature [136], empirical tests have shown that the top 3 best performing PFSP MILP models are, in this order: TS3, TBA and Wilson. In this work, we will observe that the performance obtained with the PFSP robust counterparts is rather different to the existing performance of deterministic PFSP MILP models.

4.6.3 Comparative performance of the Robust Counterpart models

Model	% Best Performance	% Solved 10x2	% Solved 10x3	% Solved 10x4	% Solved 10x5	% Solved	Avg. % gap	Median iterations	Median time
Manne	45.70	99.56	96.56	91.67	90.89	94.67	0.07	5.00	88.10
Liao-You	34.38	99.44	96.44	91.89	90.11	94.47	0.08	5.00	85.32
Wilson	13.78	99.33	95.00	83.56	80.89	88.38	2.33	5.00	276.03
TS2	6.03	99.89	95.67	85.22	85.67	90.78	3.08	4.00	436.48
TS3	4.95	99.56	92.67	78.11	86.78	89.28	1.65	5.00	334.56
TBA	0.51	99.33	87.78	71.44	69.67	82.06	1.76	5.00	883.80
WST2	0.14	98.00	86.78	69.67	66.44	80.22	2.09	5.00	1,143.74

Table 4.2 – Robust PFSP C&CG performance comparison, given all RC models and instances solved. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; % Solved $< n \times m >$ represents the percentage of solved instances of size $n \times m$; Avg. % Gap is the average percentage gap of solutions from instances not solved to optimality; Median time is the median execution time, in seconds; Median iterations is the median of the number of iterations performed.

With particular interest in examining the impact of the budget of uncertainty parameter on scheduling performance, when solving each instance, we tested the RPFS-TWCT models by varying Γ according to ten ratios (10, 20, 30, 40, 50, 60, 70, 80, 90 and 100%) of operations with uncertain processing times.

Table 4.2 summarizes the obtained results with a performance comparison of the RC models. In this table, we present medians to mitigate the effect of instances not solved within the time limit. Manne C&CG is the one that solves the majority of the instances. It also obtains the lowest average % gap for instances not solved to optimality. The *% Best Performance* measurement indicates that the Manne model solved 46% of instances with the best performance, followed by Liao-You, that solved 34%, and Wilson, with 14%. Measurements *% Solved 10 $\times$ 4* and *% Solved 10 $\times$ 5* reveal that the RC models which rely on job assignment constraints solved less instances to optimality within the time limit. The *%Solved* and *Median time* measurements also favor the dichotomous-based models.

A second and deeper analysis, grouped by instance size, presents, in Table 4.3, the average performance of each RC model, including average run time values. When using average, the results of all instances (even outliers) are taken into account. Standard deviation is also included as a secondary measure. Additionally, the average number of iterations and its standard deviation are

		Dichotomous-based		Assignment-based				
Instance size	Measure	Manne	Liao-You	Wilson	TS2	TS3	TBA	WST2
10x2	% Best Performance	36.83	17.43	31.10	12.90	0.78	1.23	0.11
	% Solved	99.56	99.44	99.33	99.89	99.56	99.33	98.00
	Avg. % gap	0.00	0.00	0.07	0.00	5.78	0.52	0.78
	Std. dev. of % gap	0.00	0.00	0.18	0.00	11.39	0.68	0.85
	Avg. iterations	7.69	8.64	5.60	3.81	4.57	3.93	3.98
	Std. dev. of iterations	56.32	66.12	19.00	3.53	16.31	4.71	2.47
	Avg. MP time	112.94	104.42	137.05	316.55	210.11	375.57	666.10
	Avg. SP time	0.85	0.95	0.38	0.39	0.55	0.51	0.37
	Avg. time	113.79	105.37	137.43	316.95	210.67	376.07	666.47
	Std. dev. of time	532.32	546.95	618.42	802.44	604.03	944.73	1,354.99
10x3	% Best Performance	49.65	32.26	6.78	6.85	8.15	0.00	0.00
	% Solved	96.56	96.44	95.00	95.67	92.67	87.78	86.78
	Avg. % gap	0.00	0.04	4.88	1.20	1.20	2.26	1.23
	Std. dev. of % gap	0.00	0.21	9.66	1.63	1.71	8.64	1.68
	Avg. iterations	18.87	17.70	6.21	6.20	6.17	5.62	5.71
	Std. dev. of iterations	76.10	67.38	7.72	7.62	5.21	3.36	4.18
	Avg. MP time	416.65	477.63	1,144.08	1,379.19	1,276.20	2,151.87	2,302.40
	Avg. SP time	7.35	9.19	8.97	8.13	4.36	5.53	3.47
	Avg. time	424.00	486.82	1,153.05	1,387.32	1,280.56	2,157.40	2,305.87
	Std. dev. of time	1,317.86	1,386.44	1,888.22	2,005.33	2,104.25	2,440.00	2,559.65
10x4	% Best Performance	30.26	60.27	10.11	2.09	4.13	0.31	0.48
	% Solved	91.67	91.89	83.56	85.22	78.11	71.44	69.67
	Avg. % gap	0.12	0.11	1.65	1.98	2.04	2.20	2.25
	Std. dev. of % gap	0.33	0.35	2.20	2.34	2.51	2.79	2.89
	Avg. iterations	23.26	24.42	10.11	7.75	7.73	6.57	6.25
	Std. dev. of iterations	69.66	77.97	18.05	6.80	6.47	3.78	3.48
	Avg. MP time	970.49	930.11	2,076.57	2,234.34	2,524.83	3,394.95	3,544.79
	Avg. SP time	52.43	46.19	48.41	43.88	53.68	34.56	28.33
	Avg. time	1,022.92	976.31	2,124.98	2,278.23	2,578.52	3,429.51	3,573.12
	Std. dev. of time	2,047.19	2,018.60	2,643.78	2,568.41	2,828.57	2,819.48	2,881.46
10x5	% Best Performance	66.95	29.10	4.40	1.04	7.09	0.32	0.00
	% Solved	90.89	90.11	80.89	85.67	86.78	69.67	66.44
	Avg. % gap	0.04	0.08	1.15	1.10	1.11	1.16	2.37
	Std. dev. of % gap	0.16	0.26	1.56	1.65	1.40	1.64	5.82
	Avg. iterations	35.26	32.64	8.23	8.03	6.72	6.79	5.93
	Std. dev. of iterations	109.28	100.17	12.70	12.17	4.90	6.61	4.75
	Avg. MP time	807.29	882.73	2,204.63	1,971.19	1,850.99	3,242.81	3,535.38
	Avg. SP time	196.24	229.35	208.22	280.31	206.78	177.76	298.72
	Avg. time	1,003.53	1,112.08	2,412.85	2,251.50	2,057.77	3,420.57	3,834.10
	Std. dev. of time	2,086.74	2,166.21	2,726.14	2,532.00	2,523.15	2,916.78	2,878.93

Table 4.3 – Robust PFSP C&CG performance comparison, for each instance size $n \times m$ and RC model. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; Avg. % Gap and Std. dev. of % Gap are the mean and standard deviation of the percentage gap of solutions from instances not solved to optimality; Avg. iterations and Std. dev. of iterations are the mean and standard deviation of the number of iterations performed; Avg. time MP(SP) is the average time to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

Model	Variable	α=10%	α=20%	α=30%	α=40%	α=50%	α^{max}=30%	α^{max}=50%	α^{max}=100%	α^{max}=200%
Manne	% Best Performance	55.10	75.00	82.14	75.00	38.89	60.61	80.21	65.98	67.01
	% Solved	98.00	96.00	85.00	73.00	73.00	99.00	96.00	100.00	98.00
	Avg. % gap	0.00	0.00	0.00	0.03	0.06	0.00	0.00	0.00	0.57
	Std. dev. of % gap	0.00	0.00	0.00	0.11	0.20	0.00	0.00	0.00	0.08
	Avg. iterations	10.02	20.02	57.25	95.02	82.81	6.71	24.29	8.82	12.36
	Avg. MP time	198.80	343.21	1,151.25	1,996.76	1,997.19	150.97	425.11	365.75	636.58
	Avg. SP time	13.34	103.91	215.03	457.65	575.86	13.88	81.86	89.68	214.94
	Avg. time	212.14	447.12	1,366.27	2,454.41	2,573.06	164.85	506.97	455.43	851.52
	Std. dev. of time	1,007.64	1,401.65	2,531.63	3,052.05	2,960.42	719.07	1,419.05	861.10	1,350.03
Liao-You	% Best Performance	40.82	22.92	17.86	23.61	58.33	25.25	16.67	27.84	32.99
	% Solved	98.00	96.00	84.00	72.00	72.00	99.00	96.00	97.00	97.00
	Avg. % gap	0.00	0.00	0.00	0.06	0.10	0.00	0.00	0.10	0.85
	Std. dev. of % gap	0.00	0.00	0.01	0.19	0.27	0.00	0.00	0.12	0.66
	Avg. iterations	10.47	17.30	49.77	86.26	80.75	5.20	23.08	8.77	12.19
	Avg. MP time	230.66	395.28	1,314.84	2,080.92	2,109.72	178.98	470.81	466.47	696.90
	Avg. SP time	19.05	120.31	270.29	537.54	640.74	13.57	98.86	110.64	253.15
	Avg. time	249.71	515.59	1,585.13	2,618.46	2,750.46	192.55	569.67	577.11	950.05
	Std. dev. of time	1,021.04	1,419.79	2,593.14	3,060.68	3,027.40	723.47	1,426.31	1,258.07	1,532.42
TS2	% Best Performance	0.00	2.06	0.00	0.00	0.00	3.03	1.10	2.33	0.00
	% Solved	100.00	97.00	83.00	75.00	70.00	99.00	91.00	86.00	70.00
	Avg. % gap	0.00	0.00	0.22	0.55	0.89	0.00	0.19	1.63	2.46
	Std. dev. of % gap	0.00	0.00	0.33	0.63	1.14	0.00	0.29	1.41	2.50
	Avg. iterations	3.76	5.99	8.20	11.61	11.20	4.76	10.37	7.47	8.94
	Avg. MP time	1,017.22	1,584.34	2,063.25	2,798.91	3,133.31	669.08	1,549.04	1,800.18	3,125.41
	Avg. SP time	197.62	354.42	496.12	465.72	542.74	15.98	73.83	126.85	249.46
	Avg. time	1,214.85	1,938.76	2,559.38	3,264.63	3,676.05	685.06	1,622.87	1,927.04	3,374.87
	Std. dev. of time	1,404.58	2,030.54	2,545.68	2,696.03	2,697.30	1,182.24	2,337.84	2,468.36	3,084.71
Wilson	% Best Performance	3.09	1.25	1.32	5.71	5.88	9.09	6.59	4.71	0.00
	% Solved	97.00	80.00	76.00	70.00	68.00	99.00	91.00	85.00	62.00
	Avg. % gap	0.07	0.34	0.68	0.79	1.17	0.00	0.23	1.93	2.18
	Std. dev. of % gap	0.12	0.34	0.55	0.76	1.10	0.00	0.37	1.77	2.46
	Avg. iterations	3.90	7.98	8.42	9.12	13.13	4.70	8.53	7.26	11.00
	Avg. MP time	1,165.72	2,266.70	2,619.72	3,019.32	3,257.35	747.88	1,508.29	1,941.32	3,315.41
	Avg. SP time	133.02	203.90	321.46	349.65	352.45	22.32	68.82	132.49	289.86
	Avg. time	1,298.74	2,470.60	2,941.17	3,368.97	3,609.79	770.20	1,577.11	2,073.81	3,605.27
	Std. dev. of time	1,944.91	2,777.29	2,764.50	2,903.50	2,836.88	1,265.93	2,319.08	2,529.15	3,098.39

Table 4.4 – Robust PFSP C&CG performance comparison for instance size 10×5, grouped by α and α^{max} values, and RC model. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; Avg. % Gap is the average percentage gap of solutions from instances not solved to optimality; Avg. iterations is the average number of iterations performed; Avg. time MP(SP) is the average time to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

Instance size	Variable	TS2	Wilson	Liao-You	Manne	Liao-You-Hybrid	Manne-Hybrid	Wilson-Hybrid
15x5	% Best Performance	0	0	0	0	9.09	37.06	56.32
	% Solved	5.71	6.9	12.98	14.15	31.78	41.67	47.33
	Avg. % gap	20.13	7.46	6.16	4.62	4.04	3.59	5.29
	Std. dev. of % gap	14.53	11.75	10.15	8.42	6.19	5.53	7.39
	Avg. iterations	2.14	3.05	3	3.31	4.76	5.91	5.83
	Std. dev. of iterations	0.73	0.87	1.23	1.49	1.85	2.48	3.63
	Avg. MP time	11,073.67	13,277.03	11,681.69	11,218.73	10,795.78	9,344.76	8,893.31
	Avg. SP time	3,050.96	358.64	1,834.73	2,101.6	640.71	988.21	920.33
	Avg. time	14,124.64	13,635.67	13,516.42	13,320.34	11,436.49	10,332.97	9,813.64
	Std. dev. of time	1,222.25	2,849.15	2,907.75	3,302.32	5,220.83	5,755.23	5,902.1

Table 4.5 – Robust PFSP C&CG performance comparison, for instance size 15×5, RC models Manne, Liao-You, Wilson and TS2, along with Hybrid C&CG models Manne-Hybrid, Liao-You-Hybrid, and Wilson-Hybrid. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit of 14400s; Avg. % Gap and Std. dev. of % Gap are the mean and standard deviation of the percentage gap of solutions from instances not solved to optimality; Avg. iterations and Std. dev. of iterations are the mean and standard deviation of the number of iterations performed; Avg. time MP(SP) is the average time to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

listed. As we could expect, these results show that, as instance size grows, the models become harder to solve, as seen on the smaller percentage of solved instances and increased average execution time. However, our results show that this is especially true for the assignment-based models.

A complementary investigation, based on the α and α^{max} parameters, is portrayed in Table 4.4. In this context, we explore solution statistics regarding the four best performing models, when solving 10×5 instances. It is possible to note the decrease of model performance as the α and α^{max} values grow. This can be observed in the *% Solved*, *Avg. % gap* and *Avg. time* rows, from columns $\alpha = 10\%$ until $\alpha = 50\%$, and from columns $\alpha^{max} = 30\%$ until $\alpha^{max} = 200\%$.

According to our experiments, the Liao-You and Manne Robust Counterparts are the ones that perform best when solving the RPFS-TWCT problem. The possible reason is related to how the objective function is calculated in each robust counterpart model. In all assignment-based models (namely Wilson, WST2, TBA, TS2 and TS3), when calculating the total weighted completion time, we multiply the weight of job i by its corresponding completion time. However, in these models, there are no variables representing the completion time of job index i. Instead, they represent the completion time of job in *position* k. To properly calculate the objective function, the adopted solution involves the creation of an auxiliary variable F_i, representing the completion time of job i (on the last machine M_m). Then, in order to calculate each value of F_i, it is necessary to apply several Big-M constraints, which make the MILP model relaxation weaker.

On the other hand, the Liao-You and Manne robust counterparts, which are based on disjunctive constraints, directly offers these variables which represent the completion time of job index i, so the only Big-M constraints in the model are the ones already present in the original model.

Finally, when solving the 15×5 instances, the largest ones in the test-bed, we perceived a drastic performance reduction of the algorithm. For this reason, besides extending the time limit parameter to $14,400$ seconds, we chose to solve these instances with the four best performing RC models so far: Manne, Liao-You, TS2 and Wilson. As shown in Table 4.5, the percentage of solved instances drops from 98% to 14% when applying the best-performing Manne model. Also, when analyzing the % gap of instances not solved to optimality, the average % gap is considerably higher in all models, as well as its variance. We can see that, for these instances, the C&CG algorithm was not able to perform more than 3 iterations on average.

4.6.4 Hybrid C&CG method performance

We will now analyse the performance of the hybrid algorithm enhancement, designed to overcome the performance limitation observed when solving the C&CG master problems.

The last three columns of Table 4.5 show additional C&CG results obtained with the new hybrid solution method. The Wilson-Hybrid model reached the best performance, for solving the highest proportion of instances to optimality (47%), and also for obtaining the shortest solution time in 56% of the instances solved to optimality, when compared to the other RC models. All hybrid solution methods achieved drastic performance improvements when compared to the initial solution method results, where the best-performing conventional C&CG algorithm, Manne, had obtained only 14% of instances solved to optimality, along with an average % gap (standard deviation of % gap) of 4.62 (8.42), respectively.

One possible answer to the obtained results may be found in how the hybridization of the solution method works. The partial permutation σ is built iteratively, using the same job fixation order of the combinatorial branch-and-bound method, i.e., fixing one job at a time, from left to right. Given this solution representation, whenever a new job k is fixed in the partial permutation, new cuts have to be added to the existing MILP model of the corresponding node in the B&B tree, in order to make the solutions from combinatorial B&B and MILP compatible. Experimental data shows that, in the case of hybrid Wilson method, the cuts added to each node, which are based on the job assignment binary variables $Z_{i,j}$, turn out to be stronger than the cuts added to the Liao-You and Manne models, which are based on the job precedence variable $D_{i,k}$.

4.7 Case study on two real instances

In this section, we assess the quality and level of robustness of scheduling solutions for two real problem instances, the first one representing a platform with 9 *oil-wells* and 4 *maintenance tasks* (9×4) and the second one a different platform with size 15×5. The processing time matrices $\overline{p}$ and $\widehat{p}$ were obtained from the available operation history. The following solution methods were used:

— **Det**: deterministic PFSP solution [149] with **P**=$\{\overline{p}_{r,i}\}, r \in \mathbb{M}, i \in \mathbb{J}$.

— **RPFS(Γ)**: Wilson-Hybrid RPFS solution method, described in Section 4.5. The Γ parameter is used to control the level of the conservativeness of the robust model, as a fraction of the number of operations $m \times n$. It varies from 5% to 100%, with 5% intervals. The robust model with $\Gamma = 0\%$ is equivalent to **Det**, while the one with $\Gamma = 100\%$ is the deterministic model that is entirely risk-averse and overestimates all parameters. The other values of Γ model intermediate risk aversions.

— **SimGRASP**: stochastic PFSP simheuristic method from [40], properly modified to find the schedule that minimizes the *expected total weighted completion time*. SimGRASP is a modified GRASP metaheuristic that incorporates Monte Carlo Simulation to solve the PFSP with random processing times. Given its stochastic nature, we obtained 25 independent runs for each instance file. Then, for result comparison purposes, for each independent run, we calculated the robust cost $\mathcal{Z}$ at each Γ protection level. Finally, we stored, for each instance, the smallest and largest robust costs found within these 25 simheuristic executions. We call them **SimGRASP-Min(25)** and **SimGRASP-Max(25)**.

We assessed the robustness of each obtained solution σ by calculating the *robust cost* at different protection levels Γ, using the worst-case MILP model defined in Section 4.4.2. Figure 4.2 depicts the *robust cost* $\mathcal{Z}(\sigma, \Gamma)$ of each solution σ under different protection levels Γ. For clarity of the graphs, the robust costs for some protection levels were omitted.

Observe that, as the protection level Γ increases, so does the robust cost, i.e., the total weighted completion time (TWCT) of the worst-case scenario defined by the protection level. In other words, higher values of Γ are equivalent to a greater quantity of operations with deviated processing times, which directly impacts the *solution cost* $\mathcal{Z}(\sigma, \Gamma)$. In the case studies from Figure 4.2, the extreme cases occur whenever $\Gamma \geq 60\%$, yielding the highest robust costs.

From the viewpoint of the decision-maker at the oil company who needs to hedge against worst-case maintenance costs, it would be preferable to obtain a solution method that performs well under different protection levels. With this in mind, in the two graphs presented, we identify which scheduling method (and respective solution) presents the best (smallest) robust cost, considering all Γ values. Regarding the first graph (9×4 instance), note that **RPFS(10)** offers improved protection against worst-case scenarios for $10\% \leq \Gamma \leq 20\%$, while **RPFS(25)** is the best-performing robust solution considering $25\% \leq \Gamma \leq 35\%$. Finally, **RPFS(50)** is indicated for higher protection levels $\Gamma \geq 45\%$. We also highlight the disappointing worst-case performance of both the nominal solution **Det** and the stochastic method. The vast distance between the robust costs of the stochastic method, i.e., **SimGRASP-Min(25)** and **SimGRASP-Max(25)**, reveals a significant exposure to the realization of worst-case scenarios, which is represented by the highlighted area in the graph.

In its turn, the larger instance (15×5) presents a distinct behavior in robust cost differences between distinct protection levels. In Figure 4.2(b), we can observe that **RPFS(50)** presents the best

overall protection against worst-case scenarios, considering $\Gamma \geq 20\%$. Once again, the solutions **Det** and **SimGRASP-Max(25)** present high robust costs. In particular, for $\Gamma = 30\%$, the robust cost provided by **RPFS(35)** is 2% cheaper than **Det** and 3% cheaper than **SimGRASP-Max(25)**.

In summary, the choice of a robust solution depends on the instance and the desired protection level. The examples above illustrate how RPFS can provide a pool of robust schedules, depending on the value of Γ. With these options, the decision-makers can choose one of the schedules based on their risk preferences. Also, remark that, if the stochastic heuristic method is chosen, depending on the solution returned by the algorithm, the worst-case performance may be weak, as can be seen on the robust costs achieved by **SimGRASP-Max(25)**. Indeed, neither **SimGRASP** nor the deterministic models have the objective of minimizing the *worst-case TWCT*.

4.7.1 Analysis based on Monte-Carlo simulation

As a complementary analysis, we evaluate the expected behavior of obtained problem solutions. The *TWCT* distribution of the obtained robust schedules was simulated by subjecting the processing time matrix to random perturbations. In particular, in each Monte Carlo simulation run, the (actual) processing time $\widetilde{p}_{r,i}, \forall r \in \mathbb{M}, i \in \mathbb{J}$, was independently drawn from a predefined probability distribution, yielding a random processing time matrix $\widetilde{P}$. For this purpose, we used lognormal, symmetric triangular, and uniform distributions in $[\overline{p} - \widehat{p}, \overline{p} + \widehat{p}]$ to generate random processing times. We generated $10,000$ processing time matrices $\widetilde{P}$. Then, for each **RPFS(Γ)** solution σ^{Γ}, obtained with a specific protection level Γ, we processed the set of all corresponding *TWCT* values $\varphi(\sigma^{\Gamma}, \widetilde{P})$ obtained through simulation on $\widetilde{P}$. The same was made for the solutions returned by **Det** and **SimGRASP-Min(25)**.

We first focus on simulation results presented in Figure 4.3(a). Regarding the 9×4 instance, we can observe that, in the long run, the TWCT performance of **RPFS(5)**, **RPFS(10)** and **RPFS(25)** are equivalent to **SimGRASP**, a method specialized at optimizing the expected TWCT. On the other hand, as stated in the worst-case analysis of Figure 4.2(a), not all schedules are sufficiently immune against worst-case scenarios. For instance, if the decision-maker assumes an intermediate protection level of $\Gamma \leq 35\%$, the two most appropriate schedules, with smallest robust costs, are **RPFS(25)** and **RPFS(10)**.

When analyzing the 15×5 instance in Figure 4.3(b), the following robust solutions present expected TWCT performance quite similar to **SimGRASP**: **RPFS(5)**, **RPFS(15)** and **RPFS(35)**. Taking the worst-case evaluation into account and considering a protection level $15\% \leq \Gamma \leq 40\%$, these three robust solutions also provide better protection against worst-case costs, when compared to the stochastic solution method.

Finally, Table 4.6 presents some statistics related to the simulation of processing times of the 9×4 instance. In this analysis, whenever the same robust solution has been obtained for more than one Γ parameter value, their (equivalent) statistics were reported in the same column. Given $10,000$

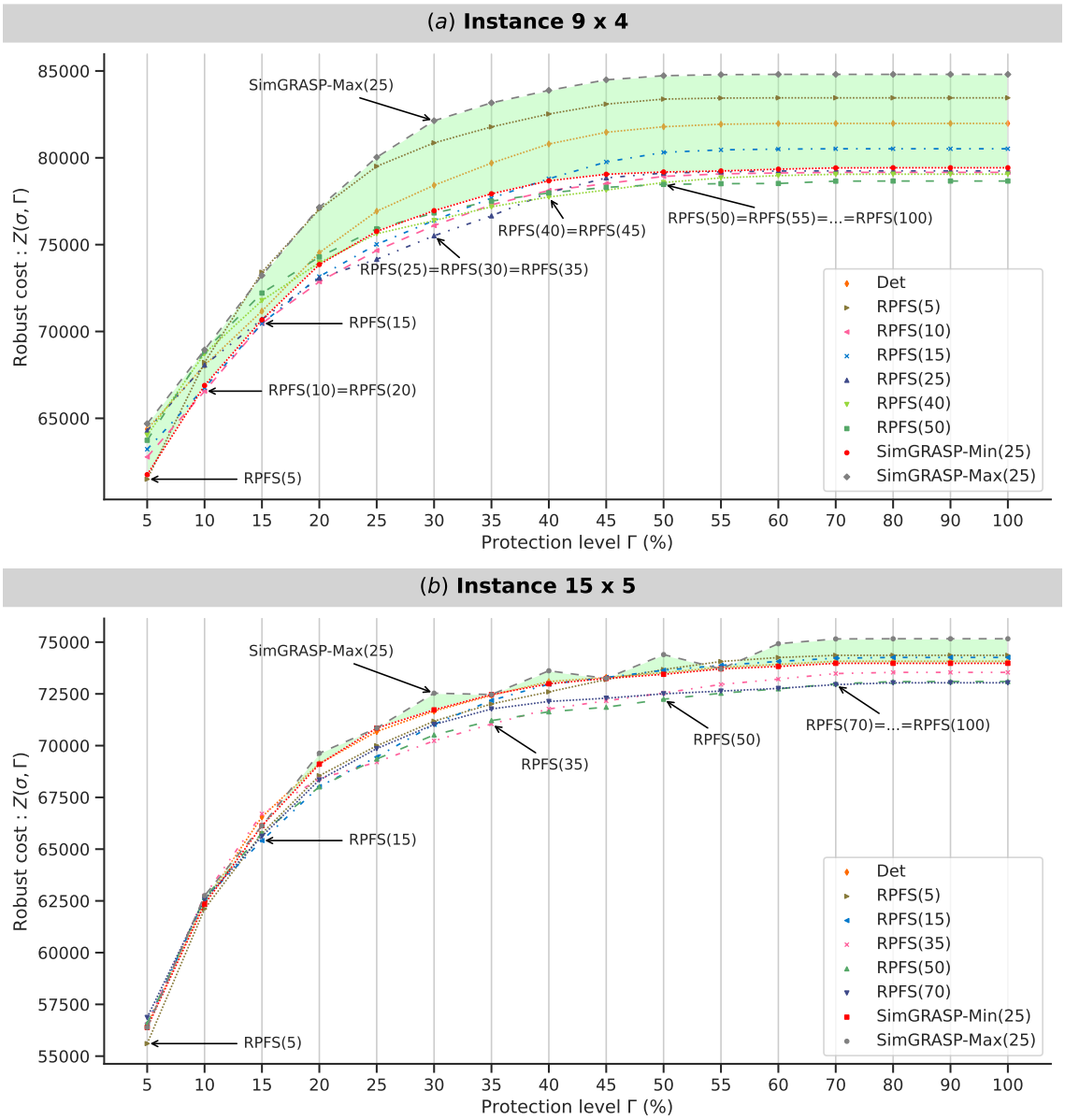

Figure 4.2 – Robust cost of deterministic, RPFS and SimGRASP solutions versus protection level Γ%. All presented RPFS solutions are optimal.

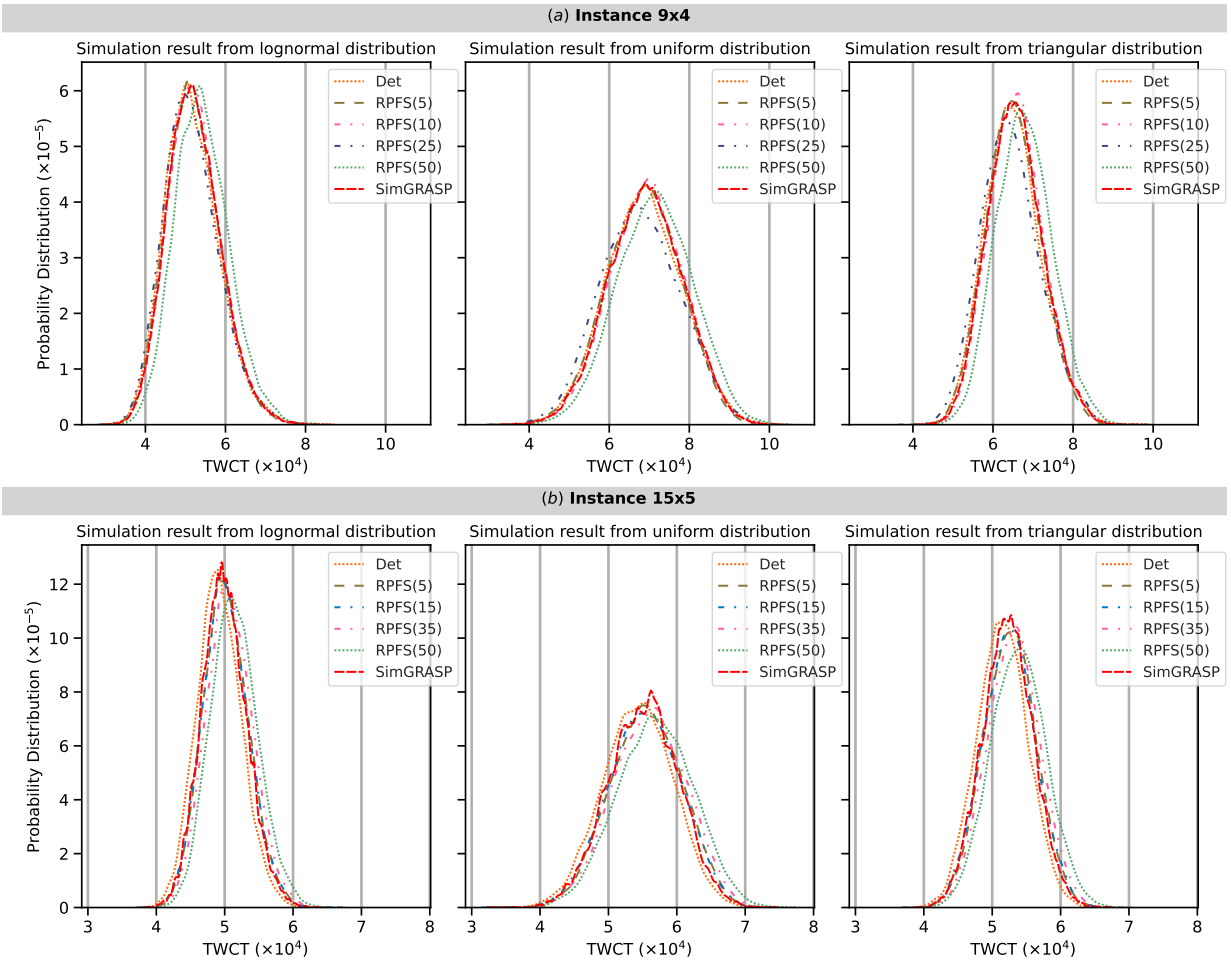

Figure 4.3 – Probability distributions of $TWCT$ value for RPFS(Γ), Det and SimGRASP solutions, according to simulation results from lognormal, triangular, and uniform distributions for uncertain processing times.

processing time matrices $\widetilde{P}$ obtained after simulation runs, let $\varphi(\sigma)$ be the random cost (*TWCT*) of scheduling σ, which depends on the realization of P. $E(\varphi(\sigma))$ and $SD(\varphi(\sigma))$ are empirical estimations of expectation and standard deviation of $\varphi(\sigma)$, respectively. Also, $VaR(\varphi(\sigma))$ and $CVaR(\varphi(\sigma))$ are the value-at-risk and conditional value-at-risk of $\varphi(\sigma)$, respectively, both at 95% confidence level. In other words, $VaR(\varphi(\sigma))$ is equivalent to the 0.95 quantile of $\varphi(\sigma)$, while $CVaR(\varphi(\sigma))$ represents the average of the largest 5% values of $\varphi(\sigma)$. Finally, $Max(\varphi(\sigma))$ is the maximum observed $\varphi(\sigma)$ in the simulation.

Observe that **RPFS(25,30,35)** has the least $E(\varphi(\sigma))$ in all distributions. When analyzing the largest observed TWCT, **RPFS(5)**, has the **lowest** $Max(\varphi(\sigma))$ for lognormal, triangular and uniform distributions. The best solutions for **Det** and **SimGRASP** did not provide minimum values for any measure of the simulated distributions. Indeed, **SimGRASP** presented the worst values for the largest observed TWCT in lognormal and triangular distributions.

Distribution	Measure	RPFS(5)	RPFS(10, 20)	RPFS(15)	RPFS(20)	RPFS(25,30,35)	RPFS(40, 45)	RPFS(50,...,100)	Det	SimGRASP
lognormal	E($\varphi(\sigma)$)	52,028	52,340	51,647	52,340	51,548	54,375	54,221	51,774	52,254
	SD($\varphi(\sigma)$)	6,619	6,606	6,820	6,606	6,941	6,749	6,772	6,841	6,725
	VaR($\varphi(\sigma)$)	63,528	63,812	63,712	63,812	63,939	66,126	65,959	63,751	63,972
	CVaR($\varphi(\sigma)$)	66,839	67,319	67,344	67,319	67,843	69,442	69,447	67,677	67,532
	Max($\varphi(\sigma)$)	79,857	82,729	83,287	82,729	82,151	83,973	82,780	84,934	86,193
triangular	E($\varphi(\sigma)$)	65,110	65,854	64,481	65,854	64,435	68,049	67,890	65,121	65,599
	SD($\varphi(\sigma)$)	6,739	6,687	7,176	6,687	7,392	6,893	6,946	7,027	6,865
	VaR($\varphi(\sigma)$)	76,531	77,215	76,973	77,215	77,362	79,567	79,620	77,399	77,372
	CVaR($\varphi(\sigma)$)	79,360	80,181	80,060	80,181	80,801	82,532	82,649	80,683	80,383
	Max($\varphi(\sigma)$)	90,901	94,283	93,286	94,283	93,362	96,123	96,399	96,200	97,623
uniform	E($\varphi(\sigma)$)	68,566	69,399	67,838	69,399	67,730	71,538	71,426	68,739	69,143
	SD($\varphi(\sigma)$)	9,153	9,066	9,706	9,066	9,878	9,331	9,414	9,432	9,280
	VaR($\varphi(\sigma)$)	83,425	84,246	83,831	84,246	84,266	86,717	86,839	84,490	84,393
	CVaR($\varphi(\sigma)$)	86,487	87,533	87,398	87,533	88,013	90,096	90,223	88,086	87,699
	Max($\varphi(\sigma)$)	98,076	100,090	100,462	100,090	102,587	102,107	102,600	101,782	102,039

Table 4.6 – Simulation summary for instance 9×4 with RPFS(Γ), Det and SimGRASP solutions after $10,000$ simulation runs under lognormal, triangular, and uniform distributions of operation processing times. Minimum and maximum values, for each row, are highlighted. Similar robust solutions for different Γ values are grouped in the same column (e.g., RPFS(10, 20)).

Also, by analyzing the smallest maximum TWCT obtained in triangular distribution simulations, the value $Max(\varphi(\sigma))$ observed for scheduling **RPFS(25,30,35)** is 4.3% cheaper than **SimGRASP**, and, at the same time, its expected TWCT is 1.8% less than the stochastic schedule. Following these observations, the decision-maker of the oil company can evaluate the hedge provided by the obtained robust solutions, and choose a specific solution (and associated protection level) that does not cause a significant increase in the expected solution cost, when compared to stochastic and deterministic solutions.

4.7.2 Evaluating hedge value and price of robustness

Given a protection level Γ, besides robust cost $\mathcal{Z}$, two other measures can be used to evaluate performance: hedge value H and price of robustness η, defined as:

$$H(\Gamma) = \mathcal{Z}(\bar{\sigma}^*, \Gamma) - \mathcal{Z}(\sigma^*_\Gamma, \Gamma), \tag{4.84}$$

$$\eta(\Gamma) = \varphi(\sigma^*_\Gamma, \overline{P}) - \varphi(\bar{\sigma}^*, \overline{P}), \tag{4.85}$$

where σ^*_Γ is the optimal solution of **RPFS(Γ)**, $\bar{\sigma}^*$ is the optimal solution of **Det(P=)**, and $\varphi(.)$ is the *TWCT* function.

The first measure, $H(\Gamma)$, represents the value gained from adopting the robust sequence σ^*_Γ, instead of the optimal nominal sequence $\bar{\sigma}^*$ in the occurrence of the worst-case scenario associated with protection level Γ. A visual interpretation of $H(\Gamma)$ can be made in Figure 4.2, by analysing to the robust cost difference between the solutions from **Det** ($\bar{\sigma}^*$) and **RPFS(Γ)** (σ^*_Γ) methods, at each protection level Γ. The second measure, $\eta(\Gamma)$ is defined as the price paid by the decision-maker for employing the robust sequence σ^*_Γ in place of the optimal nominal sequence $\bar{\sigma}^*$ in the scenario of nominal processing times (when $P = \overline{P}$, i.e., no processing time deviations). In other words, $H(\Gamma)$ can be seen as the *regret of employing sequence* $\bar{\sigma}^*$ *in the worst-case scenario*, and $\eta(\Gamma)$ represents

the *trade-off between robustness and optimality*.

	Instance Name / Measure			
	9 x 4		15 x 5	
Γ %	η %	H %	η %	H %
0	0.00%	0.00%	0.00%	0.00%
5	2.11%	4.69%	1.81%	1.62%
10	2.97%	2.24%	1.69%	0.82%
15	1.95%	1.02%	2.72%	1.75%
20	2.97%	2.30%	4.86%	1.72%
25	2.28%	3.73%	4.50%	2.33%
30	2.28%	3.85%	5.70%	1.90%
35	2.28%	3.97%	3.47%	1.95%
40	7.47%	3.92%	4.74%	1.98%
45	7.47%	4.26%	4.97%	1.87%
50	6.87%	4.23%	4.58%	1.77%
55	6.87%	4.36%	5.35%	1.63%
60	6.87%	4.39%	4.91%	1.64%
70	6.87%	4.22%	4.86%	1.55%
80	6.87%	4.22%	4.86%	1.43%
90	6.87%	4.22%	4.86%	1.42%
100	6.87%	4.22%	4.86%	1.42%

Table 4.7 – Relative robustness price $\eta(\Gamma)\%$ and hedge value $H(\Gamma)\%$ for instances 9×4 and 15×5.

Table 4.7 displays the relative price of robustness $\eta(\Gamma)\% = \frac{\varphi(\sigma^*_\Gamma,\overline{P})-\varphi(\bar{\sigma}^*,\overline{P})}{\varphi(\bar{\sigma}^*,\overline{P})}$ and hedge value $H(\Gamma)\% = \frac{\mathcal{Z}(\bar{\sigma}^*,\Gamma)-\mathcal{Z}(\sigma^*_\Gamma,\Gamma)}{\mathcal{Z}(\sigma^*_\Gamma,\Gamma)}$ for various protection levels, based on the two instances from this case study. Among the schedules obtained when solving RPFS with different Γ levels, the best ones, which maximize hedge value $H(\Gamma)\%$, are **RPFS(5)** for instance 9×4, and **RPFS(25)** for instance 15×5.

Based on the simulation framework presented in Section 4.7.1, we close this section with a further analysis of the actual cost overhead of robust solutions in the long run. Two performance measures are calculated for instance 9×4, as shown in Table 4.8. The obtained results show that, for the protection levels Γ used in the study, robust solutions **RPFS(15)**, **RPFS(25)**, **RPFS(30)**, and **RPFS(35)** present two important characteristics: (i) high proportion of cheapest solutions ($\omega(\Gamma) > 50\%$), and (ii) smaller expected costs, i.e., negative relative cost difference $\Delta\eta(\Gamma)$. All in all, RPFS provides a pool of robust schedules decision-makers can choose based on their risk preferences.

4.8 Discussion

In this chapter, we presented an exact solution method for the m-machine robust permutation flow shop problem to minimize the *total weighted completion time*. Similarly to the previous chap-

	Distribution / Measure					
	lognormal		triangular		uniform	
Γ %	ω	$\Delta\eta$	ω	$\Delta\eta$	ω	$\Delta\eta$
5	44.6%	0.5%	49.7%	0.0%	52.6%	-0.3%
10	34.5%	1.1%	31.3%	1.1%	36.0%	1.0%
15	53.9%	-0.2%	68.5%	-1.0%	67.4%	-1.3%
20	34.5%	1.1%	31.3%	1.1%	36.0%	1.0%
25	52.2%	-0.4%	62.0%	-1.1%	63.6%	-1.5%
30	52.1%	-0.4%	62.0%	-1.1%	63.6%	-1.5%
35	52.2%	-0.4%	62.0%	-1.1%	63.6%	-1.5%
40	13.5%	5.0%	10.8%	4.5%	17.1%	4.1%
45	13.5%	5.0%	10.9%	4.5%	17.1%	4.1%
50	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
55	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
60	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
70	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
80	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
90	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
100	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%

Table 4.8 – Simulation results for instance 9×4, for different protection levels $\Gamma \in \{5\%, 15\%, \ldots, 100\%\}$. Comparison is based on two measures: (i) $\omega(\Gamma)$ is the % of simulated scenarios (over a total of 10,000) where RPFS(Γ) obtained smaller $TWCT$ cost when compared to Det(P=); (ii) $\Delta\eta(\Gamma) = Avg_{\lambda \in \mathbb{S}}\left[\frac{\varphi(\sigma^*_\Gamma, P^\lambda) - \varphi(\bar{\sigma}^*, P^\lambda)}{\varphi(\bar{\sigma}^*, P^\lambda)}\right]$ is the average relative cost difference between RPFS(Γ) and Det(P=), given all simulated scenarios λ.

ters, the models developed here are based on budgeted uncertainty, a powerful tool for handling robustness and the trade-off between cost and risk, avoiding the over-conservativeness of the conventional robust scheduling approaches.

Besides proposing a set of benchmark instances for the problem, we developed seven robust-counterpart formulations, coupled with an exact solution method based on Column-and-Constraint Generation (C&CG). Additionally, we implemented a hybrid C&CG method which relies on two strategies to enhance the processing of larger problem instances. First, a branching strategy used in the combinatorial branch-and-bound for scheduling problems. Second, a new lower bound for the robust problem, based on an existing bound used in the deterministic case. Computational experiments suggest that the improved algorithm can handle test problems with $n \leq 15$, reaching the same instance-size limit of the best-performing deterministic solution methods. Despite the longer average processing time required to solve the larger 15×5 instances, good-quality solutions were obtained by the hybrid C&CG based on the Wilson formulation, with 47% of the instances solved to optimality. For the remaining instances, average gaps of 5% were obtained.

We have also assessed the cost of the solutions returned by the robust model and compared them to deterministic and stochastic solutions. According to simulations based on three probability distributions, our solution method was capable of protecting against worst-case scenarios, with just a small overhead in the expected solution cost.

Experimental results indicate the feasibility of applying this robust solution method to real-world problem instances, such as the ones from the oil and gas industry, whose current solutions are obtained through methods that disregard either uncertainty or the impact of worst-case scenarios. Based on their risk preferences, decision-makers can then choose an appropriate schedule from a pool of robust solutions, with different levels of exposure to uncertainty.

Chapter 5

Conclusions and perspectives

The solution methods to all robust problems studied in this thesis rely on the budgeted uncertainty set, which provides a pool of robust solutions under different levels of realization of uncertainty, to hedge against worst-case costs. The great advantage of the applied budgeted approach lies in its capacity to avoid the over-conservativeness of conventional robust optimization approaches, at the same time enabling the decision-maker to evaluate the trade-off between robustness and higher solution cost.

5.1 Conclusions

The first research work of this thesis is related to the Contract Collaboration Problem (CCP) and microgrid energy scheduling. We proposed a comprehensive robust framework for flexible bilateral energy contract engagement, coupled with a Real-Time Command Strategy (RTCS), suited for energy management of microgrids with uncertainty in both production and consumption of energy. The framework is backed by a robust optimization model under budgeted uncertainty, whose solution provides a cost-effective contract commitment planning for a given time horizon, while minimizing the worst-case microgrid cost. It also features a set of control strategies for real-time energy trading and scheduling.

Concerning the CCP framework, a case study was conducted on a real microgrid, with four problem instances generated, one for each season of the year. Monte-Carlo simulations were applied to assess the performance of different robust model solutions under distinct protection levels. When used as input for real-time energy scheduling strategies, each robust solution was compared to three deterministic solution alternatives. Based on a set of real-world-inspired energy purchase contracts, simulation results have confirmed the efficacy of different robust-based RTCS strategies on specific protection levels, depending on the scenario type. The obtained results show that the effectiveness of each robust solution (and associated protection level) largely depends on the microgrid's load profile and renewable production. These, in turn, may vary according to the season of the year or even according to other recurrent patterns. Due to scope limitations, we opted not to deepen our analysis based on specific dates of the year, but we believe additional studies could be performed on this subject.

It is also worth noting that the CCP contract framework, although not yet a reality for small consumers, has the potential to be used as a tool to improve the predictability of consumption and production of microgrids. Moreover, as microgrids and actively managed distribution networks become widespread, the introduction of forward energy trading is expected to happen at the local level [23]. In this sense, the CCP framework could be used in the context of small aggregators of prosumers, improving energy exchange between them.

The second part of the thesis explored robust scheduling problems based on the budgeted uncertainty set. We first studied the 2-machine robust permutation flow shop (2RPFS) problem (*makespan* objective), developing an exact solution method based on Column-and-Constraint Generation (C&CG) techniques. To this end, together with new robust-counterpart formulations, we employed a worst-case determination procedure for the problem, using polynomial-time dynamic programming. Extensive experimental results demonstrated the effectiveness in obtaining optimal robust schedules for small and medium-sized problems (e.g., instances with $n \leq 100$). Moreover, we developed a case study with two representative instances and assessed the trade-off between solution quality and cost, comparing robust solutions to deterministic and stochastic ones. Based on Monte-Carlo simulations performed on three probability distributions, we observed that the obtained robust schedules are very competitive for specific budget parameter values, with no overhead in the expected solution cost.

The next step of the research involved the development of solution methods for the m-machine robust permutation flow shop problem to minimize the worst-case *makespan*. Firstly, we extended the same exact solution method previously applied to the 2RPFS problem. New robust counterpart formulations were developed, along with a m-machine generalization of the two-machine worst-case procedure based on dynamic programming. Nonetheless, despite our best efforts, the exact method was unable to efficiently solve even the smallest 100-job Taillard-based instances (20×5). Therefore, to allow the solution of larger problem instances, we designed and implemented a Greedy Randomized Adaptive Search Procedure (GRASP) metaheuristic, with the worst-case procedure incorporated as the objective function. Extensive experimental results demonstrated that, compared to the baseline C&CG exact method, the GRASP algorithm was very efficient in obtaining robust schedules. In the absence of literature instances, four sets of benchmark test-beds were proposed, including instances based on the classical flow shop literature. After evaluating both C&CG and GRASP algorithms on this test-bed, GRASP was shown to produce optimal solutions on most small and medium-sized instances. Indeed, GRASP obtained a higher proportion of solved instances than the exact method, which evidences its efficiency and solution quality.

The research conducted with these robust scheduling methods allowed us to develop, in the final study, an exact solution method for the m-machine robust permutation flow shop problem to minimize the worst-case *total weighted completion time* (RPFS-TWCT), a problem with direct application in the oil and gas industry. Similar to the previous studies, since no set of problem instances

was available in the literature, we also proposed a set of benchmark instances to assess the algorithm's performance. After extensive computational experiments and algorithm enhancements, we prepared a case study to evaluate the robust solutions obtained from problem instances that depict the maintenance effort of a real oil platform. Once again, we applied Monte-Carlo simulations to compare the solutions returned by the robust, deterministic, and stochastic methods. With a particular interest in the expected behavior of the solutions from each method, we wanted to verify a possible increase in the expected solution cost of the robust method in the long run. According to simulations based on three probability distributions, the hedge provided by the obtained robust solutions did not cause a significant increase in the expected solution cost compared to the stochastic and deterministic solutions.

In summary, experimental results indicated the feasibility of applying the proposed robust solution methods to real-world problem instances, including those from the oil and gas industry. Additionally, the same methodology applied in our studies can be followed by decision-makers, when choosing an appropriate model result. By examining a pool of solutions with distinct budgeted uncertainty levels, they can select the most appropriate one according to their risk preferences, thus balancing solution protection and expected performance in the long run.

5.2 Perspectives

The different investigations conducted in this work suggest diverse questions that, to the best of our knowledge, remain open. At the end of each chapter, we have already discussed the most technical aspects related to these questions. In the following, we review the larger research points deserving some longer-term efforts.

When evaluating CCP models, we only tested contracts where the microgrid buys electricity from the energy companies, but not the opposite (i.e., selling contracts). In this sense, the energy exchange between multiple microgrids could be seen as a game-theoretic model. It would also be interesting to perform sensitivity analysis on energy contract values, such as fixed and variable prices, and minimum and maximum amounts. The main question to be answered would be: at which point does one contract become more attractive than the other?

The methodology itself could also be enhanced. For example, extending the RTCS method to use forecasting techniques that help predict the amount of energy produced or consumed by the microgrid, according to the season of the year, or following other trends and patterns. Several machine learning algorithms and frameworks are available for this task [122]. Another useful technique would be Reinforcement Learning [147], which could also prove helpful inside the RTCS, making better decisions on the order of execution of energy-related operations. One option involves the incorporation of an online learning algorithm for real-time energy scheduling. For instance, the scheduling procedure could predict which kind of operation would be better suited at a given period (e.g., store or sell

surplus energy; retrieve from storage or buy from engaged contracts) and in which order.

Lastly, model refinements could be included, such as adding ramping constraints for generators and improving the battery efficiency and degradation model. Furthermore, the uncertainty set adopted in the robust optimization model could also be extended to take into account the temporal correlations of uncertainty, following a similar approach to the dynamic uncertainty set adopted by [92]. The main idea is that, in any time slot, individual oscillations in energy production and consumption values are not independent of each other, but correlated.

Regarding the robust scheduling problems studied in this thesis, the computational complexity of 2RPFS under budgeted uncertainty remains an important topic to be studied. Future research may also attempt to develop efficient heuristics for the m-machine RPFS-TWCT problem. However, this largely depends on the efficient solution to the worst-case problem. In this sense, a computational complexity study is needed, and, if possible, a pseudopolynomial algorithm for the worst-case TWCT could be designed. Given that the RPFS-TWCT problem is NP-hard, this would be particularly important. To this end, our exact approach could play a useful role in the performance evaluation of any heuristic.

Another avenue for future research could involve the design of new sets of robust problem instances, structured according to each problem's characteristics (e.g., where processing times are such that the critical path remains fixed or alternates according to a pattern). This would allow a deeper analysis of each robust problem's easiest and most difficult instances. Moreover, the solution methods proposed in this work could be extended to similar scheduling problems and different objective functions, such as total tardiness and total flow time.

Finally, concerning all problems studied in this thesis, an interesting topic involves the generation of multiple optimal solutions for the robust problem and how to inspect the quality of each one. As [115] noted, different robust solutions may even have the same *worst-case* objective value, but might still differ on the *mean* objective value, or the expected value in the long run, considering the realization of several possible scenarios. Therefore, an important research question would be: given a protection level Γ, is it possible to generate a handful of optimal solutions and compare them in terms of the hedge provided as well as the expected performance, so that the decision-maker can choose the solution that best meets these requirements? Such analysis might be conducted as an additional step inside the framework applied in this thesis, and, in this sense, the work of [61] on Pareto Robustly Optimal solutions [1] could serve as a starting point.

1. A solution is called Pareto Robustly Optimal (PRO) if there is no other robustly feasible solution that has better objective value for at least one scenario, and for all other scenarios in the uncertainty set the objective value is not worse [61].

Appendices

Appendix A

Optimization criteria for Robust Permutation Flow Shop (RPFS) Problem

Different optimization criteria may be used to choose a robust solution [46, 2]. The first and simplest criterion is the *minimax* (also known as the *absolute robust* criterion). In this case, given a minimization problem, the robust decision is made by choosing a solution that minimizes the highest cost over all possible scenarios.

A second possible criterion is called *minimax* regret, which aims to find the least maximum regret over all possible scenarios. Regret can be either defined as the *difference* or the *ratio* between the resulting cost of the candidate decision and the cost of the decision that would have been taken if uncertain input data were known in advance (prior to the decision time). In the first case, where regret is defined as the difference between two values, the so-called *robust deviation* decision is obtained. For the latter case (regret being the ratio of two values), the resulting decision is the *relative robust* decision. Either way, decisions originated from these two criteria tend to be less conservative than the absolute robust criterion, because they measure the amount of deviation of a specific candidate decision by comparing its performance with the performance of the "ideal" solution of each possible scenario (i.e. its corresponding optimal decision if uncertainty was known ahead of time).

Regarding the **Rob-PFS** problem, existing works deal with two possible measures of robustness, associated with the *makespan* objective: *minimax makespan* and *minimax regret makespan*.

A.1 Minimax Makespan (MM) Rob-PFS problem with 2 machines

The objective of the Minimax Makespan is to *minimize* the *maximum* possible **makespan** associated with a schedule.

The *absolute robust schedule* (σ_{ARS}) satisfies:

$$\max_{\lambda \in \Lambda}\{\varphi(\sigma_{ARS}, \mathbf{P}^{\lambda})\} = \min_{\sigma \in \Sigma} \max_{\lambda \in \Lambda}\{\varphi(\sigma, \mathbf{P}^{\lambda})\} \tag{A.1}$$

A.2 Minimax Regret Makespan (MRM) Rob-PFS problem

In this case, the objective is to *minimize* the *maximum* possible **regret** associated with a schedule. Given a scenario $\lambda \in \Lambda$, the **regret** of a schedule σ is measured as the absolute difference between:

1. the makespan of the schedule σ for the scenario λ [$\varphi(\sigma, \mathbf{P}^{\lambda})$];
2. the makespan of the *optimal* schedule σ^*_{λ} for the scenario λ [$\varphi(\sigma^*_{\lambda}, \mathbf{P}^{\lambda})$].

The so-called *absolute deviation robust schedule* (σ_{ADRS}) satisfies [74]:

$$\max_{\lambda \in \Lambda}\{\varphi(\sigma_{ADRS}, \mathbf{P}^{\lambda}) - \varphi(\sigma^*_{\lambda}, \mathbf{P}^{\lambda})\} = \min_{\sigma \in \Sigma} \max_{\lambda \in \Lambda}\{\varphi(\sigma, \mathbf{P}^{\lambda}) - \varphi(\sigma^*_{\lambda}, \mathbf{P}^{\lambda})\} \tag{A.2}$$

The research on minimax regret optimization models is well-established nowadays. In particular, most of the works on flow shop robust scheduling rely on this measure of robustness, as seen in the literature review. Since the PFSP is NP-hard for 3 or more machines, it is not possible to obtain exact solutions to the MRM RPFS problem when $m \geq 3$, given the difficulty to calculate the optimal deterministic schedule for a given scenario.

Appendix B

A BRIEF INTRODUCTION TO THE BUDGETED UNCERTAINTY MODEL

Bertsimas and Sim [16] proposed an alternative method of representing data uncertainty, avoiding the complication of non-linear formulations.

Consider the following nominal linear optimization problem:

$$\text{Maximize } c^T x \tag{B.1}$$

$$\text{subject to } Ax \leq B \tag{B.2}$$

$$x \geq 0 \tag{B.3}$$

In the above formulation, we assume that data uncertainty only affects the elements in matrix A. Without loss of generality, it can be assumed that the objective function c is not subject to uncertainty, since it is possible to use the objective maximize z, and add the constraint $z \leq c^T x$, and then include this constraint into $Ax \leq B$.

Consider a specific row i of matrix $A = [a_{ij}]$ and let N_i represent the set of coefficients in row i that are subject to uncertainty. Each entry a_{ij}, $j \in N_i$ is modeled as a symmetric and bounded random variable $\tilde{a}_{ij}$, $j \in N_i$ [15], and takes values in the interval $[a_{ij} - \tilde{a}_{ij}, a_{ij} + \tilde{a}_{ij}]$. Associated with the uncertain data $\tilde{a}_{ij}$, we define the random variable $\eta_{ij} = (a_{ij} - \tilde{a}_{ij})/\hat{a}_{ij}$, which obeys an unknown but symmetric distribution, taking values in $[-1, 1]$.

Given the i-th constraint of the nominal problem $a_i^T x \leq b_i$, now consider that the set of coefficients a_{ij}, $j \in N_i$ take values according to a symmetric distribution *with mean equal to the nominal value* $\overline{a}_{ij}$ *in the interval* $[\overline{a}_{ij} - \hat{a}_{ij}, \overline{a}_{ij} + \hat{a}_{ij}]$. For every i, a parameter Γ_i (not necessarily integer) is introduced, and takes values in the interval $[0, \mid N_i \mid]$.

Given these premises, Bertsimas and Sim defined the following symmetric and bounded uncertainty data set, in which an uncertain budget parameter Γ_i is introduced to control the degree of uncertainty:

$$U_i = \{a_{ij} \mid a_{ij} \in [\overline{a}_{ij} - \hat{a}_{ij}, \overline{a}_{ij} + \hat{a}_{ij}], \forall j \in N_i; \sum_{j \in N_i} \frac{|a_{ij} - \overline{a}_{ij}|}{\hat{a}_{ij}} \leq \Gamma_i\}, \ \forall i \tag{B.4}$$

Constraint (B.4) indicates that, at the same time, up to $\lfloor \Gamma_i \rfloor$ uncertainty coefficients can get

their worst-case values, in which $\Gamma_i \in [0, \mid N_i \mid]$ and is not necessarily an integer. The idea is that all uncertain coefficients in N_i are often unlikely to simultaneously reach their worst-case values. Therefore, it may be adequate to protect against up to $\lfloor \Gamma_i \rfloor$ of those coefficients in N_i reaching their worst-case values, and one coefficient, say t_i, to change by $(\Gamma_i - \lfloor \Gamma_i \rfloor)\ \hat{a}_{i,t_i}$. Regarding this uncertainty set, a protection function $\beta(x,\ \Gamma_i)$ for each constraint i is defined as follows:

$$\beta(x,\ \Gamma_i) = \max_{\left\{\widetilde{N}_i \cup \{t_i\} : \widetilde{N}_i \subseteq N_i, |\widetilde{N}_i| = \lfloor \Gamma_i \rfloor, t_i \in N_i \setminus \widetilde{N}_i\right\}} \{\sum_{j \in N_i} \hat{a}_{ij} x_j + (\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{i,t_i} x_{t_i}\},\ \forall i \tag{B.5}$$

where $x = (x_j,\ \forall j)$ is the decision variable vector. Based on the above idea, the budgeted RO model (still nonlinear) is presented as:

$$\text{(Budget-RO)}\ \ \text{Maximize} \sum_j c_j x_j \tag{B.6}$$

$$\text{subject to} \sum_j a_{ij} x_j + \beta(x,\ \Gamma_i) \leq b_i,\ \ \forall i \tag{B.7}$$

$$x_j \geq 0,\ \ \forall j \tag{B.8}$$

The expression (B.6) to be maximized is the objective function. Constraints (B.7) are the functional constraints, which use a protection function $\beta(x, \Gamma_i)$ for each constraint to control the degree of uncertainty. Constraints (B.8) are the non-negativity constraints. Note that by varying $\Gamma_i \in [0, \mid N_i \mid]$, the decision maker is able to adjust the solution robustness against the level of solution conservatism. When $\Gamma_i = 0, \beta(x,\ \Gamma_i) = 0$, the problem is equivalent to the nominal problem. On the other hand, if $\Gamma_i = N_i$, we obtain the most conservative version of the problem, which is equal to the worst-case model, as developed by Soyster [128].

In order to solve the *Budget-RO* model, Bertsimas and Sim applied the duality theory, reformulating the problem as an LP problem. They began by formulating an LP problem whose objective function equals the protection function $\beta(x, \Gamma_i)$. In other words, given the decision variable vector x, the protection function $\beta(x,\ \Gamma_i)$ defined in Constraint (B.5) is equal to the objective function of the following LP problem, which will be called *LP-Beta(i)*.

$$\text{[LP-Beta(i)]}\ \ \text{Maximize} \sum_{j \in N_i} \hat{a}_{ij} x_j y_{ij} \tag{B.9}$$

$$\text{subject to} \sum_{j \in N_i} y_{ij} \leq \Gamma_i \tag{B.10}$$

$$0 \leq y_{ij} \leq\ 1,\ \ \forall j \in N_i \tag{B.11}$$

The expression (B.9) defines an objective function equivalent to the protection function. Constraints (B.10) and (B.11) ensure that $\lfloor \Gamma_i \rfloor$ of those coefficients in N_i reach their worst-case values and one coefficient to change by $(\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{i,t_i}$. This is equivalent to selecting a subset:

$\{\widetilde{N_i} \cup \{t_i\} \quad : \quad \widetilde{N_i} \subseteq N_i, \mid \widetilde{N_i} \mid = \lfloor \Gamma_i \rfloor, t_i \in N_i \setminus \widetilde{N_i}\}$ with the corresponding cost function: $\sum_{j \in N_i} \hat{a}_{ij} x_j + (\Gamma_i - \lfloor \Gamma_i \rfloor \hat{a}_{i,t_i} x_{t_i})$.

Let ω_i and π_{ij} $(\forall j \in N_i)$ be the dual variables corresponding to constraints (B.10) and (B.11), respectively. The dual problem *[DLP-Beta(i)]* is obtained as follows:

$$\text{[DLP-Beta(i)]} \quad \text{Minimize} \sum_{j \in N_i} \pi_{ij} + \Gamma_i \, \omega_i \tag{B.12}$$

$$\text{subject to } \omega_i + \pi_{ij} \geq \hat{a}_{ij} x_j, \quad \forall j \in N_i \tag{B.13}$$

$$\pi_{ij} \geq 0, \quad \forall j \in N_i \tag{B.14}$$

$$\omega_i \geq 0 \tag{B.15}$$

Since *LP-Beta(i)* is feasible and bounded for $\Gamma_i \in [0, \mid N_i \mid]$, by strong duality, *DLP-Beta(i)* is also feasible and bounded, and both models have the same objective function value. The protection function $\beta(x, \Gamma_i)$ is therefore equal to the objective function value of *DLP-Beta(i)*.

Finally, by substituting *DLP-Beta(i)* (in place of $\beta(x, \Gamma_i)$) into *Budget-RO*, the following LP model is obtained:

$$\text{(LP-Budget-RO)} \quad \text{Maximize} \sum_j c_j x_j \tag{B.16}$$

$$\text{subject to } \sum_j a_{ij} x_j + \Gamma_i \omega_i + \sum_{j \in N_i} \pi_{ij} \leq b_i, \quad \forall i \tag{B.17}$$

$$\omega_i + \pi_{ij} \geq \hat{a}_{ij} x_j, \quad \forall i, \forall j \in N_i \tag{B.18}$$

$$\pi_{ij} \geq 0, \quad \forall i, \forall j \in N_i \tag{B.19}$$

$$\omega_i \geq 0, \quad \forall i \tag{B.20}$$

$$x_j \geq 0, \quad \forall j \tag{B.21}$$

Appendix C

Proof of 2RPFS worst-case scenario extreme points

Although not being explicitly stated, it is assumed that the 2RPFS budget parameters Γ_1 and Γ_2 are both integers.

Now suppose that, given a permutation σ, we are calculating its worst-case scenario λ and worst-case makespan $\mathcal{Z}(\sigma)$, with fractional budget values of Γ_1 and Γ_2. In this case, the processing time deviation amount $\delta_{r,i}$ lies in the $[0, 1]$ interval. Every worst-case scenario has an associated critical path as well as a set of operations $O_{r,i}$ that belong to this critical path. We propose the following lemma.

Lemma C.0.1. *Given a permutation σ with worst-case scenario λ, consider a machine M_r, for which there are two operations O_{r,i_1} and O_{r,i_2} that belong to the critical path and $0 < \delta^{\lambda}_{r,i_1}, \delta^{\lambda}_{r,i_2} < 1$. Also suppose, without loss of generality, that $\widehat{p}_{r,i_1} \leq \widehat{p}_{r,i_2}$. Then, there is an equivalent worst-case scenario $\lambda^{\#}$ whose makespan is greater or equal than the makespan of scenario λ, such that at least one of the following is true: $\delta^{\lambda^{\#}}_{r,i_1} = 0$ or $\delta^{\lambda^{\#}}_{r,i_2} = 1$.*

The proof follows by construction. The idea is that, given two operations O_{r,i_1} and O_{r,i_2} that belong to the critical path, it is always possible to "transfer" a portion of the processing time deviation $\delta_{r,i}$ from the operation with largest maximum deviation (O_{r,i_2}) to the shortest one (O_{r,i_1}), until at least one of their $\delta_{r,i}$ values reaches an extreme value of zero or one.

Let $\lambda^{\#}$ be the worst-case scenario constructed in a way that at least $\delta^{\lambda^{\#}}_{r,i_1} = 0$ or $\delta^{\lambda^{\#}}_{r,i_2} = 1$:

$$\delta^{\lambda^{\#}}_{r,i} \equiv \begin{cases} \delta^{\lambda}_{r,i_1} - \min(\delta^{\lambda}_{r,i_1}, 1 - \delta^{\lambda}_{r,i_2}) & \text{if } i = i_1 \\ \delta^{\lambda}_{r,i_2} + \min(\delta^{\lambda}_{r,i_1}, 1 - \delta^{\lambda}_{r,i_2}) & \text{if } i = i_2 \\ \delta^{\lambda}_{r,i} & \text{otherwise} \end{cases} \tag{C.1}$$

Note that, since operations O_{r,i_1} and O_{r,i_2} belong to the critical path in both scenarios λ and $\lambda^{\#}$, the makespan φ of the alternative worst-case scenario $\lambda^{\#}$ is calculated as:

$$\varphi(\lambda^{\#}) = \varphi(\lambda) - (\delta^{\lambda}_{r,i_1}\widehat{p}_{r,i_1} + \delta^{\lambda}_{r,i_2}\widehat{p}_{r,i_2}) + (\delta^{\lambda^{\#}}_{r,i_1}\widehat{p}_{r,i_1} + \delta^{\lambda^{\#}}_{r,i_2}\widehat{p}_{r,i_2})$$

$$\varphi(\lambda^{\#}) = \varphi(\lambda) - (\delta^{\lambda}_{r,i_1}\widehat{p}_{r,i_1} + \delta^{\lambda}_{r,i_2}\widehat{p}_{r,i_2}) +$$
$$+ (\delta^{\lambda}_{r,i_1} - \min(\delta^{\lambda}_{r,i_1}, 1 - \delta^{\lambda}_{r,i_2}))\widehat{p}_{r,i_1} + (\delta^{\lambda}_{r,i_2} + \min(\delta^{\lambda}_{r,i_1}, 1 - \delta^{\lambda}_{r,i_2}))\widehat{p}_{r,i_2}$$

$$\varphi(\lambda^{\#}) = \varphi(\lambda) + \min(\delta^{\lambda}_{r,i_1}, 1 - \delta^{\lambda}_{r,i_2})(\widehat{p}_{r,i_2} - \widehat{p}_{r,i_1})$$

And the following holds true:

$\varphi(\lambda^{\#}) - \varphi(\lambda) = \min(\delta^{\lambda}_{r,i_1}, 1 - \delta^{\lambda}_{r,i_2})(\widehat{p}_{r,i_2} - \widehat{p}_{r,i_1}) \geq 0$, since $\widehat{p}_{r,i_2} \geq \widehat{p}_{r,i_1}$.

We just proved that the makespan in scenario $\lambda^{\#}$ is greater or equal than the makespan in scenario λ. □

Remark that the same argument as above is valid for pairs of jobs that do not belong to the critical path. In particular, a change in their processing times does not affect the (worst-case) makespan.

Now we can prove the following.

Corollary C.0.2. *Given a permutation σ with worst-case scenario λ, for every machine M_r, there is an equivalent worst-case scenario $\lambda^{\#}$ where all $\delta^{\lambda^{\#}}_{r,i}$ are either zero or one, except for at most one operation $O_{r,i'}$ where $\delta^{\lambda^{\#}}_{r,i'} = \Gamma_r - \lfloor \Gamma_r \rfloor$.*

The proof follows by contradiction. Given a worst-case scenario λ induced by a permutation σ, let $x(\lambda, r)$ be the number of operations $O_{r,i}$ on machine M_r such that $0 < \delta^{\lambda}_{r,i} < 1$. Suppose, by contradiction, that $x(\lambda, r) \geq 2$. Therefore, by successively applying Lemma C.0.1, we end up an alternative worst-case scenario $\lambda^{\#}$, such that $x(\lambda^{\#}, r) \leq 1$ and, by Equation (C.1), $\delta^{\lambda^{\#}}_{r,i'} = \Gamma_r - \lfloor \Gamma_r \rfloor$. □

With that being said, it is possible to have the budget parameters Γ_1 and Γ_2 fractional, with a slight adaptation in the dynamic programming table, taking into account that, for each machine M_r, exactly one of its operations will deviate its processing time by a factor of $\Gamma_r - \lfloor \Gamma_r \rfloor$.

BIBLIOGRAPHY

[1] Agostinho Agra, Marielle Christiansen, Rosa Figueiredo, Lars Magnus Hvattum, Michael Poss, and Cristina Requejo, « The robust vehicle routing problem with time windows », *in*: *Computers & Operations Research* 40.*3* (2013), pp. 856–866 (cit. on pp. 12, 31).

[2] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten, « Min-max and min-max regret versions of combinatorial optimization problems: A survey », *in*: *European Journal of Operational Research* 197.*2* (2009), pp. 427–438, ISSN: 03772217, DOI: 10.1016/j.ejor.2008.09.012 (cit. on pp. 58, 60, 136).

[3] Ali Allahverdi, Harun Aydilek, and Asiye Aydilek, « Single machine scheduling problem with interval processing times to minimize mean weighted completion time », *in*: *Computers and Operations Research* 51 (2014), pp. 200–207, ISSN: 03050548, DOI: 10.1016/j.cor.2014.06.003, URL: http://dx.doi.org/10.1016/j.cor.2014.06.003 (cit. on p. 98).

[4] Maurice Allais, *Méthode d'évaluation des perspectives économiques de la recherche minière sur de grands espaces: application au Sahara algérien*, la Loire, 1956 (cit. on p. 13).

[5] Didier Aussel, Luce Brotcorne, Sébastien Lepaul, and Léonard von Niederhäusern, « A trilevel model for best response in energy demand-side management », *in*: *European Journal of Operational Research* 281.*2* (2020), pp. 299–315 (cit. on p. 20).

[6] Igor Averbakh, « The minmax regret permutation flow-shop problem with two jobs », *in*: *European Journal of Operational Research* 169.*3* (2006), pp. 761–766, ISSN: 03772217, DOI: 10.1016/j.ejor.2004.07.073 (cit. on p. 59).

[7] F Babonneau, J-P Vial, and R Apparigliato, « Robust optimization for environmental and energy planning », *in*: *Uncertainty and Environmental Decision Making*, Springer, 2009, pp. 79–126 (cit. on p. 12).

[8] Kenneth R Baker, *Introduction to sequencing and scheduling*, John Wiley & Sons, 1974 (cit. on p. 55).

[9] Kenneth R. Baker and Dan Trietsch, « Three heuristic procedures for the stochastic, two-machine flow shop problem », *in*: *Journal of Scheduling* 14.*5* (2011), pp. 445–454, ISSN: 10946136, DOI: 10.1007/s10951-010-0219-4 (cit. on p. 58).

[10] J. Balasubramanian and I. E. Grossmann, « A novel branch and bound algorithm for scheduling flowshop plants with uncertain processing times », *in*: *Computers and Chemical Engineering* 26.*1* (2002), pp. 41–57, ISSN: 00981354, DOI: 10.1016/S0098-1354(01)00735-9 (cit. on p. 58).

[11] Y. Bassok, A. Bixby, R. Srinivasan, and H. Z. Wiesel, « Design of Component-supply Contract with Commitment Revision Flexibility », *in*: *IBM Journal of Research and Development* 41.*6* (1997), pp. 693–704 (cit. on p. 26).

[12] A. Ben-Tal et al., « Retailer-Supplier Flexible Commitments Contracts: A Robust Optimization Approach », *in*: *Manufacturing & Service Operations Management* 7.*3* (2005), pp. 248–271 (cit. on p. 26).

[13] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski, « Adjustable robust solutions of uncertain linear programs », *in*: *Mathematical Programming* 99.*2* (2004), pp. 351–376 (cit. on p. 33).

[14] Aharon Ben-Tal and Arkadi Nemirovski, « Robust optimization–methodology and applications », *in*: *Mathematical Programming* 92.*3* (2002), pp. 453–480 (cit. on pp. 12, 51, 97).

[15] Aharon Ben-Tal and Arkadi Nemirovski, « Robust solutions of linear programming problems contaminated with uncertain data », *in*: *Mathematical programming* 88.*3* (2000), pp. 411–424 (cit. on pp. 61, 101, 138).

[16] Dimitris Bertsimas and Melvyn Sim, « The Price of Robustness », *in*: *Operations Research* 52.*1* (2004), pp. 35–53, ISSN: 0030-364X, DOI: 10.1287/opre.1030.0065 (cit. on pp. 14, 21, 31, 52, 60–62, 77, 95, 98, 101, 138).

[17] S Binato, WJ Hery, DM Loewenstern, and MAURICIO GC Resende, « A GRASP for job shop scheduling », *in*: *Essays and surveys in metaheuristics*, Springer, 2002, pp. 59–79 (cit. on p. 81).

[18] Philip Börjesson and Patrik Larsson, « Cost models for battery energy storage systems », Bachelor's Thesis, KTH, Energy Technology, 2018, p. 31 (cit. on p. 40).

[19] Marin Bougeret, Artur Alves Pessoa, and Michael Poss, « Robust scheduling with budgeted uncertainty », *in*: *Discrete Applied Mathematics* 261 (2019), pp. 93–107, ISSN: 0166218X, DOI: 10.1016/j.dam.2018.07.001, URL: https://doi.org/10.1016/j.dam.2018.07.001 (cit. on p. 101).

[20] O. Braun, T. C. Lai, G. Schmidt, and Y. N. Sotskov, « Stability of Johnson's schedule with respect to limited machine availability », *in*: *International Journal of Production Research* 40.*17* (2002), pp. 4381–4400, ISSN: 00207543, DOI: 10.1080/00207540210159527 (cit. on p. 58).

[21] Jacques Carlier and Ismaıl Rebaı, « Two branch and bound algorithms for the permutation flow shop problem », *in*: *European Journal of Operational Research* 90.*2* (1996), pp. 238–251 (cit. on p. 53).

[22] Hugh M Cartwright and Robert A Long, « Simultaneous optimization of chemical flowshop sequencing and topology using genetic algorithms », *in*: *Industrial & engineering chemistry research* 32.*11* (1993), pp. 2706–2713 (cit. on p. 51).

[23] E Cazalet, P De Marini, J Price, E Woychik, and J Caldwell, « Transactive energy models », *in*: *NIST transactive energy challenge: business and regulatory models working group* (2016), pp. 1–44 (cit. on pp. 20, 132).

[24] Chia-Shin Chung, James Flynn, and Omer Kirca, « A branch and bound algorithm to minimize the total flow time for m-machine permutation flowshop problems », *in*: *International Journal of Production Economics* 79.*3* (2002), pp. 185–196 (cit. on pp. 100, 116).

[25] C M Colson and M Hashem Nehrir, « Comprehensive Real-Time Microgrid Power Management and Control With Distributed Agents », *in*: *IEEE Transactions on Smart Grid* 4.*1* (Mar. 2013), pp. 617–627, ISSN: 1949-3053, DOI: 10.1109/TSG.2012.2236368 (cit. on pp. 24, 25).

[26] Carlos Adrian Correa-Florez, Andrea Michiorri, and Georges Kariniotakis, « Comparative analysis of adjustable robust optimization alternatives for the participation of aggregated residential prosumers in electricity markets », *in*: *Energies* 12.*6* (2019), p. 1019 (cit. on pp. 27, 31).

[27] Emily Craparo et al., « A robust optimization approach to hybrid microgrid operation using ensemble weather forecasts », *in*: *Applied Energy* 201 (2017), pp. 135–147, ISSN: 03062619, DOI: 10.1016/j.apenergy.2017.05.068 (cit. on p. 25).

[28] Michał Ćwik and Jerzy Józefczyk, « Evolutionary Algorithm for Minmax Regret Flow-Shop Problem », *in*: *Management and Production Engineering Review* 6.*3* (2015), pp. 3–9, DOI: 10.1515/mper-2015-0021 (cit. on pp. 59, 60).

[29] Michał Ćwik and Jerzy Józefczyk, « Heuristic algorithms for the minmax regret flow-shop problem with interval processing times », *in*: *Central European Journal of Operations Research* 26.*1* (2018), pp. 215–238, ISSN: 16139178, DOI: 10.1007/s10100-017-0485-8 (cit. on pp. 59, 60).

[30] DE Deal, Taeyong Yang, and S Hallquist, « Job scheduling in petrochemical production: two-stage processing with finite intermediate storage », *in*: *Computers & chemical engineering* 18.*4* (1994), pp. 333–344 (cit. on p. 51).

[31] Bajis Dodin, « Determining the optimal sequences and the distributional properties of their completion times in stochastic flow shops », *in*: *Computers and Operations Research* 23.*9* (1996), pp. 829–843, ISSN: 03050548, DOI: 10.1016/0305-0548(95)00083-6 (cit. on p. 58).

[32] Lingjie Duan and Rui Zhang, « Dynamic contract to regulate energy management in microgrids », *in*: *2013 IEEE International Conference on Smart Grid Communications, SmartGridComm 2013* (2013), pp. 660–665, DOI: 10.1109/SmartGridComm.2013.6688034 (cit. on pp. 20, 25).

[33] EDF, *EDF - Fiche descriptive de l'offre «Tarif Bleu»*, (accessed June 16, 2021), 2021, URL: https://particulier.edf.fr/content/dam/2-Actifs/Documents/Offres/Grille_prix_Tarif_Bleu.pdf (cit. on p. 40).

[34] Salah E. Elmaghraby and Kristin A. Thoney, « The two-machine stochastic flowshop problem with arbitrary processing time distributions », *in*: *IIE Transactions (Institute of Industrial Engineers)* 31.*5* (1999), pp. 467–477, ISSN: 15458830, DOI: 10.1080/07408179908969849 (cit. on p. 58).

[35] Gijs van Essen, Maarten Zandvliet, Paul Van den Hof, Okko Bosgra, Jan-Dirk Jansen, et al., « Robust waterflooding optimization of multiple geological scenarios », *in*: *SPE Journal* 14.*01* (2009), pp. 202–210 (cit. on p. 12).

[36] Mohammad Javad Feizollahi and Hadi Feyzollahi, « Robust quadratic assignment problem with budgeted uncertain flows », *in*: *Operations Research Perspectives* 2 (2015), pp. 114–123, ISSN: 22147160, DOI: 10.1016/j.orp.2015.06.001 (cit. on p. 67).

[37] Thomas A Feo and Mauricio GC Resende, « Greedy randomized adaptive search procedures », *in*: *Journal of global optimization* 6.*2* (1995), pp. 109–133 (cit. on p. 81).

[38] Miguel A Fernández Pérez, Fabricio Oliveira, and Silvio Hamacher, « Optimizing Workover Rig Fleet Sizing and Scheduling Using Deterministic and Stochastic Programming Models », *in*: *Industrial & engineering chemistry research* 57.*22* (2018), pp. 7544–7554 (cit. on p. 97).

[39] Victor Fernandez-Viagas, Rubén Ruiz, and Jose M Framinan, « A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation », *in*: *European Journal of Operational Research* 257.*3* (2017), pp. 707–721 (cit. on p. 81).

[40] Daniele Ferone, Paola Festa, Aljoscha Gruler, and Angel A. Juan, « Combining simulation with a GRASP metaheuristic for solving the permutation flow-shop problem with stochastic processing times », *in*: *2016 Winter Simulation Conference (WSC)*, IEEE, Dec. 2016, pp. 2205–2215, ISBN: 978-1-5090-4486-3, DOI: 10.1109/WSC.2016.7822262, URL: http://ieeexplore.ieee.org/document/7822262/ (cit. on pp. 58, 70, 123).

[41] R. Figueiredo, A. Jouglet, and D. Savoure, « Robust Control Command Strategies in a Contract-Based Collaboration Framework », *in*: *22st International Symposium on Mathematical Programming (ISMP 2015)*, Pittsburgh, EUA, Aug. 2015, p. 139 (cit. on p. 12).

[42] Jose M. Framinan and Paz Perez-Gonzalez, « On heuristic solutions for the stochastic flowshop scheduling problem », *in*: *European Journal of Operational Research* 246.*2* (2015), pp. 413–420, ISSN: 03772217, DOI: 10.1016/j.ejor.2015.05.006 (cit. on p. 58).

[43] Jose M. Framinan, Paz Perez-Gonzalez, and Victor Fernandez Viagas Escudero, « The value of real-time data in stochastic flowshop scheduling: A simulation study for makespan », *in*: *Proceedings - Winter Simulation Conference*, 2018, pp. 3299–3310, ISBN: 9781538634288, DOI: 10.1109/WSC.2017.8248047 (cit. on p. 58).

[44] Michael R Garey, David S Johnson, and Ravi Sethi, « The complexity of flowshop and jobshop scheduling », *in*: *Mathematics of operations research* 1.*2* (1976), pp. 117–129 (cit. on pp. 53, 100).

[45] Ludo F Gelders and Narayanasamy Sambandam, « Four simple heuristics for scheduling a flow-shop », *in*: *The International Journal of Production Research* 16.*3* (1978), pp. 221–231 (cit. on p. 100).

[46] A. Gerodimos, P. Kouvelis, and G. Yu, « Robust Discrete Optimization and Its Applications. », *in*: *The Journal of the Operational Research Society* 49.*12* (Dec. 1998), p. 1303, ISSN: 01605682, DOI: 10.2307/3010157, URL: https://www.springer.com/la/book/9780792342915%20https://www.jstor.org/stable/3010157?origin=crossref (cit. on pp. 60, 136).

[47] Eliana M. González-Neira and Jairo R. Montoya-Torres, « A GRASP meta-heuristic for the hybrid flowshop scheduling problem », *in*: *Journal of Decision Systems* 26.*3* (2017), pp. 294–306, DOI: 10.1080/12460125.2017.1351863 (cit. on p. 81).

[48] Eliana María González-Neira, Jairo R. Montoya-Torres, and David Barrera, « Flow-shop scheduling problem under uncertainties: Review and trends », *in*: *International Journal of Industrial Engineering Computations* 8.*4* (2017), pp. 399–426, ISSN: 19232934, DOI: 10.5267/j.ijiec.2017.2.001 (cit. on p. 57).

[49] Bram L. Gorissen, Ihsan Yanikoğlu, and Dick den Hertog, « A practical guide to robust optimization », *in*: *Omega (United Kingdom)* 53 (2015), pp. 124–137, ISSN: 03050483, DOI: 10.1016/j.omega.2014.12.006, eprint: 1501.02634 (cit. on p. 32).

[50] Ronald L Graham, Eugene L Lawler, Jan Karel Lenstra, and AHG Rinnooy Kan, « Optimization and approximation in deterministic sequencing and scheduling: a survey », *in*: *Annals of discrete mathematics*, vol. 5, Elsevier, 1979, pp. 287–326 (cit. on pp. 53, 99).

[51] Amir H Hajimiragha, Claudio A Canizares, Michael W Fowler, Somayeh Moazeni, and Ali Elkamel, « A robust optimization approach for planning the transition to plug-in hybrid electric vehicles », *in*: *IEEE Transactions on Power Systems* 26.*4* (2011), pp. 2264–2274 (cit. on p. 12).

[52] Nicholas G Hall and Chelliah Sriskandarajah, « A survey of machine scheduling problems with blocking and no-wait in process », *in*: *Operations research* 44.*3* (1996), pp. 510–525 (cit. on p. 51).

[53] Daniel P Heyman and Matthew J Sobel, *Stochastic models in operations research: stochastic optimization*, vol. 2, Courier Corporation, 1982 (cit. on pp. 11, 51).

[54] Johnny C Ho and Yih-Long Chang, « A new heuristic for the n-job, M-machine flow-shop problem », *in*: *European Journal of Operational Research* 52.*2* (1991), pp. 194–202 (cit. on p. 98).

[55] T.-P. Hong and T.-N. Chuang, « Fuzzy Gupta Scheduling for Flow Shops with More than two Machines », *in*: *International Journal of Computers and Applications* 27.*3* (2005), pp. 169–177, ISSN: 1206-212X, DOI: 10.1080/1206212X.2005.11441765 (cit. on p. 58).

[56] Ali Hooshmand et al., « Stochastic model predictive control method for microgrid management », *in*: *2012 IEEE PES Innovative Smart Grid Technologies (ISGT)*, IEEE, Jan. 2012, pp. 1–7, ISBN: 978-1-4577-2159-5, DOI: 10.1109/ISGT.2012.6175660 (cit. on p. 25).

[57] Wuhua Hu, Ping Wang, and Hoay Beng Gooi, « Toward optimal energy management of microgrids via robust two-stage optimization », *in*: *IEEE Transactions on Smart Grid* 9.*2* (2018), pp. 1161–1174, ISSN: 19493053, DOI: 10.1109/TSG.2016.2580575 (cit. on pp. 24, 25).

[58] Yingsong Huang et al., « Adaptive electricity scheduling in microgrids », *in*: *IEEE Transactions on Smart Grid* 5.*1* (2014), pp. 270–281, ISSN: 19493053, DOI: 10.1109/TSG.2013.2282823, arXiv: arXiv:1301.0528v1 (cit. on p. 25).

[59] Akhtar Hussain et al., « Optimal operation of tri-generation microgrids considering demand uncertainties », *in*: *International Journal of Smart Home* 10.*10* (2016), pp. 131–144, ISSN: 19754094, DOI: 10.14257/ijsh.2016.10.10.13 (cit. on p. 25).

[60] Akhtar Hussain et al., « Robust optimization-based scheduling of multi-microgrids considering uncertainties », *in*: *Energies* 9.*4* (2016), ISSN: 19961073, DOI: 10.3390/en9040278 (cit. on p. 25).

[61] Dan A Iancu and Nikolaos Trichakis, « Pareto efficiency in robust optimization », *in*: *Management Science* 60.*1* (2014), pp. 130–147 (cit. on p. 134).

[62] Hisao Ishibuchi, Shinta Misaki, and Hideo Tanaka, « Modified simulated annealing algorithms for the flow shop sequencing problem », *in*: *European Journal of Operational Research* 81.*2* (1995), pp. 388–398 (cit. on p. 53).

[63] Rabih A Jabr, « Adjustable robust OPF with renewable energy sources », *in*: *IEEE Transactions on Power Systems* 28.*4* (2013), pp. 4742–4751 (cit. on p. 33).

[64] ZH Jin, K Ohno, T Ito, and SE Elmaghraby, « Scheduling hybrid flowshops in printed circuit board assembly lines », *in*: *Production and Operations Management* 11.*2* (2002), pp. 216–230 (cit. on p. 51).

[65] Carlee Joe-Wong, Soumya Sen, Sangtae Ha, and Mung Chiang, « Optimized day-ahead pricing for smart grids with device-specific scheduling flexibility », *in*: *IEEE Journal on Selected Areas in Communications* 30.*6* (2012), pp. 1075–1085, ISSN: 07338716, DOI: 10.1109/JSAC.2012.120706 (cit. on p. 20).

[66] Selmer Martin Johnson, « Optimal two-and three-stage production schedules with setup times included », *in*: *Naval Research Logistics (NRL)* 1.*1* (1954), pp. 61–68 (cit. on p. 53).

[67] Edward F.Stafford Jr and Fan T. Tseng, « On the Srikar-Ghosh MILP model for the N x M SDST flowshop problem », *in*: *International Journal of Production Research* 28.*10* (1990), pp. 1817–1830, ISSN: 1366588X, DOI: 10.1080/00207549008942836 (cit. on pp. 56, 62, 102, 107).

[68] Angel A. Juan, Helena R. Lourenço, Manuel Mateo, Rachel Luo, and Quim Castella, « Using iterated local search for solving the flow-shop problem: Parallelization, parametrization, and randomization issues », *in*: *International Transactions in Operational Research* 21.*1* (2014), pp. 103–126, DOI: https://doi.org/10.1111/itor.12028 (cit. on p. 85).

[69] Philip Kaminsky and David Simchi-Levi, « Probabilistic Analysis and Practical Algorithms for the Flow Shop Weighted Completion Time Problem », *in*: *Operations Research* 46.*6* (Dec. 1998), pp. 872–882, ISSN: 0030-364X, DOI: 10.1287/opre.46.6.872, URL: http://pubsonline.informs.org/doi/abs/10.1287/opre.46.6.872 (cit. on p. 98).

[70] Philip Kaminsky and David Simchi-Levi, « Probabilistic analysis and practical algorithms for the flow shop weighted completion time problem », *in*: *Operations Research* 46.*6* (Dec. 1998), pp. 872–882, ISSN: 0030364X, DOI: 10.1287/opre.46.6.872 (cit. on p. 100).

[71] Adam Kasperski, Adam Kurpisz, and Paweł Zieliński, « Approximating a two-machine flow shop scheduling under discrete scenario uncertainty », *in*: *European Journal of Operational Research* 217.*1* (2012), pp. 36–43, ISSN: 03772217, DOI: 10.1016/j.ejor.2011.08.029 (cit. on p. 59).

[72] Mohsen Khorasany and Reza Razzaghi, « Microgrids and Local Markets », *in*: *Microgrids: Advances in Operation, Control, and Protection* (2021), pp. 151–177 (cit. on p. 26).

[73] Jamin Koo, Kyusang Han, and En Sup Yoon, « Integration of CCS, emissions trading and volatilities of fuel prices into sustainable energy planning, and its robust optimization », *in*: *Renewable and Sustainable Energy Reviews* 15.*1* (2011), pp. 665–672 (cit. on p. 12).

[74] Panos Kouvelis, Richard L Daniels, and George Vairaktarakis, « Robust scheduling of a two-machine flow shop with uncertain processing times », *in*: *Iie Transactions* 32.*5* (2000), pp. 421–432 (cit. on pp. 59, 137).

[75] Talel Ladhari and Mohamed Haouari, « A computational study of the permutation flow shop problem based on a tight lower bound », *in*: *Computers and Operations Research* 32.*7* (2005), pp. 1831–1847, ISSN: 03050548, DOI: 10.1016/j.cor.2003.12.001 (cit. on p. 53).

[76] BJ Lageweg, Jan Karel Lenstra, and AHG Rinnooy Kan, « A general bounding scheme for the permutation flow-shop problem », *in*: *Operations Research* 26.*1* (1978), pp. 53–67 (cit. on p. 114).

[77] Dipak Laha and Uday K. Chakraborty, « An efficient stochastic hybrid heuristic for flowshop scheduling », *in*: *Engineering Applications of Artificial Intelligence* 20.*6* (2007), pp. 851–856, ISSN: 09521976, DOI: 10.1016/j.engappai.2006.10.003 (cit. on p. 58).

[78] T. C. Lai and Y. N. Sotskov, « Sequencing with uncertain numerical data for makespan minimisation », *in*: *Journal of the Operational Research Society* 50.*3* (1999), pp. 230–243, ISSN: 14769360, DOI: 10.1057/palgrave.jors.2600690 (cit. on p. 58).

[79] Tsung Chyan Lai, Yuri N. Sotskov, Natalja G. Egorova, and Frank Werner, « The optimality box in uncertain data for minimising the sum of the weighted job completion times », *in*: *International Journal of Production Research* 56.*19* (2018), pp. 6336–6362, ISSN: 1366588X, DOI: 10.1080/00207543.2017.1398426 (cit. on p. 98).

[80] R.H. Lasseter and P. Paigi, « Microgrid: a conceptual solution », *in*: *2004 IEEE 35th Annual Power Electronics Specialists Conference (IEEE Cat. No.04CH37551)*, vol. 1998-June, June, IEEE, 1998, pp. 4285–4290, ISBN: 0-7803-8399-0, DOI: 10.1109/PESC.2004.1354758 (cit. on p. 19).

[81] A Leiras, S Hamacher, and A Elkamel, « Petroleum refinery operational planning using robust optimization », *in*: *Engineering Optimization* 42.*12* (2010), pp. 1119–1131 (cit. on p. 12).

[82] Mario Levorato, Rosa Figueiredo, and Yuri Frota, « Exact solutions for the two-machine robust flow shop with budgeted uncertainty », *in*: *European Journal of Operational Research* 300.*1* (2022), pp. 46–57, ISSN: 0377-2217, DOI: https://doi.org/10.1016/j.ejor.2021.10.021 (cit. on pp. 16, 110).

[83] Mario Levorato, Rosa Figueiredo, and Yuri Frota, « Robust microgrid energy trading and scheduling under budgeted uncertainty », *in*: *Expert Systems with Applications* (2022), ISSN: 0957-4174, DOI: https://doi.org/10.1016/j.eswa.2022.117471 (cit. on p. 16).

[84] Mario Levorato, Rosa Figueiredo, Yuri Y. Frota, Antoine Jouglet, and David Savourey, « Real-time command strategies for smart grids based on the Robust Contract-based Collaboration Problem », *in*: *International Network Optimization Conference (INOC 2019)*, Avignon, France, June 2019, URL: https://hal.archives-ouvertes.fr/hal-02187032 (cit. on p. 16).

[85] Mario Levorato, Rosa Figueiredo, Yuri Y. Frota, Antoine Jouglet, and David Savourey, « Real-time energy scheduling for microgrids based on the Contract Collaboration Problem », *in*: *21e congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF 2020)*, Montpellier, France, Feb. 2020, URL: https://hal.archives-ouvertes.fr/hal-02528081 (cit. on p. 16).

[86] Mario Levorato, Rosa Figueiredo, Yuri Y. Frota, and David Sotelo, « The 2-machine robust flow shop problem under budgeted uncertainty », *in*: *22eme Conferénce ROADEF de la société Française de Recherche Opérationnelle et d'Aide à la Décision*, Mulhouse, France, Apr. 2021, URL: https://hal.archives-ouvertes.fr/hal-03236596 (cit. on p. 17).

[87] Mario Levorato, Rosa Figueiredo, Yuri Frota, Antoine Jouglet, and David Savourey, « Real-time command strategies for smart grids based on the Robust Contract-based Collaboration Problem », *in*: *International Network Optimization Conference-INOC 2019*, 2019 (cit. on pp. 21, 33).

[88] Jie Li, Ruth Misener, and Christodoulos A Floudas, « Scheduling of crude oil operations under demand uncertainty: A robust optimization framework coupled with global optimization », *in*: *AIChE Journal* 58.*8* (2012), pp. 2373–2396 (cit. on p. 12).

[89] Ching-Jong Liao and Chii-Tsuen You, « An improved formulation for the job-shop scheduling problem », *in*: *Journal of the Operational Research Society* 43.*11* (1992), pp. 1047–1054 (cit. on pp. 57, 62, 102, 108).

[90] Manuel López-Ibáñez, Jérémie Dubois-Lacoste, Leslie Pérez Cáceres, Mauro Birattari, and Thomas Stützle, « The irace package: Iterated racing for automatic algorithm configuration », *in*: *Operations Research Perspectives* 3 (2016), pp. 43–58 (cit. on p. 86).

[91] Alvaro Lorca and Xu Andy Sun, « Multistage robust unit commitment with dynamic uncertainty sets and energy storage », *in*: *IEEE Transactions on Power Systems* 32.*3* (2016), pp. 1678–1688 (cit. on pp. 27, 31, 33).

[92] Álvaro Lorca and Xu Andy Sun, « Adaptive Robust Optimization With Dynamic Uncertainty Sets for Multi-Period Economic Dispatch Under Significant Wind », *in*: *IEEE Transactions on Power Systems* 30.*4* (2015), pp. 1702–1713, ISSN: 08858950, DOI: 10.1109/TPWRS.2014.2357714, arXiv: arXiv:1409.2936v2 (cit. on p. 134).

[93] Chung-Cheng Lu, Kuo-Ching Ying, and Shih-Wei Lin, « Robust single machine scheduling for minimizing total flow time in the presence of uncertain processing times », *in*: *Computers Industrial Engineering* 74 (2014), pp. 102–110, ISSN: 0360-8352, DOI: https://doi.org/10.1016/j.cie.2014.04.013 (cit. on p. 67).

[94] Alan S Manne, « On the job-shop scheduling problem », *in*: *Operations Research* 8.*2* (1960), pp. 219–223 (cit. on pp. 56, 57, 107).

[95] Wojciech Mitkowski, Janusz Kacprzyk, Krzysztof Oprzdkiewicz, and Paweł Skruch, « Blocks for the flow shop scheduling problem with uncertain parameters », *in*: *Advances in Intelligent Systems and Computing* 577 (2017), pp. 703–711, ISSN: 21945357, DOI: 10.1007/978-3-319-60699-6 (cit. on p. 58).

[96] Sanjoy Mitter, Munther Dahleh, and Mardavij Roozbehani, « Dynamic Pricing and Stabilization of Supply and Demand in Modern Power Grids », *in*: *Smart Grid Communications (SmartGridComm)* (2010), pp. 543–548 (cit. on p. 20).

[97] Shigeji Miyazaki and Noriyuki Nishiyama, « Analysis for minimizing weighted mean flow-time in flow-shop scheduling », *in*: *Journal of the Operations Research Society of Japan* 23.*2* (1980), pp. 118–133 (cit. on p. 100).

[98] N. Mladenović and P. Hansen, « Variable neighborhood search », *in*: *Computers & Operations Research* 24.*11* (1997), pp. 1097–1100 (cit. on p. 83).

[99] S. Mohammadi et al., « Optimally operating microgrids in the presence of electric vehicles and renewable energy resources », *in*: *Smart Grid Conference, SGC 2015 Sgc* (2017), pp. 66–72, DOI: 10.1109/SGC.2015.7857392 (cit. on p. 25).

[100] Thomas Morstyn, Alexander Teytelboym, and Malcolm D. McCulloch, « Bilateral contract networks for peer-to-peer energy trading », *in*: *IEEE Transactions on Smart Grid* 10.*2* (2019), pp. 2026–2035, ISSN: 19493053, DOI: 10.1109/TSG.2017.2786668 (cit. on p. 20).

[101] Viswanath Nagarajan and Maxim Sviridenko, « Tight bounds for permutation flow shop scheduling », *in*: *Mathematics of Operations Research* 34.*2* (2009), pp. 417–427 (cit. on pp. 53, 100).

[102] Apurva Narayan and Kumaraswamy Ponnambalam, « Risk-averse stochastic programming approach for microgrid planning under uncertainty », *in*: *Renewable Energy* 101 (2017), pp. 399–408 (cit. on p. 21).

[103] Muhammad Nawaz, E. Emory Enscore, and Inyong Ham, « A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem », *in*: *Omega* 11.*1* (1983), pp. 91–95, ISSN: 03050483, DOI: 10.1016/0305-0483(83)90088-9 (cit. on p. 60).

[104] Tu A. Nguyen and M. L. Crow, « Stochastic Optimization of Renewable-Based Microgrid Operation Incorporating Battery Operating Cost », *in*: *IEEE Transactions on Power Systems* 31.*3* (2016), pp. 2289–2296, ISSN: 08858950, DOI: 10.1109/TPWRS.2015.2455491 (cit. on pp. 24, 25).

[105] Pol Olivella-Rosell et al., « Optimization problem for meeting distribution system operator requests in local flexibility markets with distributed energy resources », *in*: *Applied Energy* 210 (2018), pp. 881–895, ISSN: 03062619, DOI: 10.1016/j.apenergy.2017.08.136 (cit. on p. 26).

[106] Stig Ottesen et al., « Demand side operational flexibility - A holistic stochastic optimization model for flexible consumers and prosumers », *in*: *IET Conference Publications* 2013.*615 CP* (2013), pp. 10–13, DOI: 10.1049/cp.2013.0954 (cit. on pp. 24, 25).

[107] Chao-Hsien Pan, « A study of integer programming formulations for scheduling problems », *in*: *International Journal of Systems Science* 28.*1* (1997), pp. 33–41 (cit. on p. 57).

[108] Jordi Pereira, « The robust (minmax regret) single machine scheduling with interval processing times and total weighted completion time objective », *in*: *Computers and Operations Research* 66 (2016), pp. 141–152, ISSN: 03050548, DOI: 10.1016/j.cor.2015.08.010 (cit. on p. 98).

[109] Michael L Pinedo, *Scheduling: theory, algorithms, and systems*, Springer, 2016 (cit. on pp. 53, 98, 99).

[110] Marcus Poggi and David Sotelo, « A linear time approximation algorithm for permutation flow shop scheduling », *in*: *Theoretical Computer Science* 416 (2012), pp. 87–94 (cit. on p. 53).

[111] CN Potts, « An adaptive branching rule for the permutation flow-shop problem », *in*: *European Journal of Operational Research* 5.*1* (1980), pp. 19–25 (cit. on p. 53).

[112] Chandrasekharan Rajendran and Hans Ziegler, « An efficient heuristic for scheduling in a flowshop to minimize total weighted flowtime of jobs », *in*: *European Journal of Operational Research* 103.*1* (1997), pp. 129–138 (cit. on pp. 98, 100).

[113] Glaydston Mattos Ribeiro, Geraldo Regis Mauri, and Luiz Antonio Nogueira Lorena, « A simple and robust Simulated Annealing algorithm for scheduling workover rigs on onshore oil fields », *in*: *Computers & Industrial Engineering* 60.*4* (2011), pp. 519–526 (cit. on p. 97).

[114] Paul A. Rubin, *OR in an OB World*, (accessed February 1, 2022), 2014, URL: https://orinanobworld.blogspot.com/2014/10/multiple-children-again.html (cit. on p. 114).

[115] Frans JCT de Ruiter, Ruud Brekelmans, and Dick den Hertog, « The impact of the existence of multiple adjustable robust solutions », *in*: *Mathematical Programming* 160.*1* (2016), pp. 531–545 (cit. on p. 134).

[116] José Luis Ruiz Duarte and Neng Fan, « Operations of a microgrid with renewable energy integration and line switching », *in*: *Energy Systems* 10.*2* (2019), pp. 247–272, ISSN: 18683975, DOI: 10.1007/s12667-018-0286-8 (cit. on p. 25).

[117] José Luis Ruiz Duarte, Neng Fan, and Tongdan Jin, « Multi-process production scheduling with variable renewable integration and demand response », *in*: *European J. of Operational Research* 281.*1* (2020), pp. 186–200, ISSN: 03772217, DOI: 10.1016/j.ejor.2019.08.017 (cit. on pp. 65, 110).

[118] Rubén Ruiz and Thomas Stützle, « A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem », *in*: *European Journal of Operational Research* 177.*3* (2007), pp. 2033–2049, ISSN: 03772217, DOI: 10.1016/j.ejor.2005.12.009 (cit. on p. 84).

[119] I. Sabuncuoglu and S. Goren, « Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research », *in*: *International Journal of Computer Integrated Manufacturing* 22.*2* (2009), pp. 138–157, ISSN: 13623052, DOI: 10.1080/09511920802209033 (cit. on p. 51).

[120] Yakov Shafransky and Viktor Shinkarevich, « On the complexity of constructing a minmax regret solution for the two-machine flow shop problem under the interval uncertainty », *in*: *Journal of Scheduling* 23.*6* (2020), pp. 745–749 (cit. on p. 59).

[121] Pierluigi Siano, « Demand response and smart grids – A survey », *in*: *Renewable and sustainable energy reviews* 30 (2014), pp. 461–478 (cit. on p. 22).

[122] Julien Siebert, Janek Groß, and Christof Schroth, « A Systematic Review of Packages for Time Series Analysis », *in*: *Engineering Proceedings*, vol. 5, 1, Multidisciplinary Digital Publishing Institute, 2021, p. 22 (cit. on p. 133).

[123] Marcin Siepak, « Scatter Search based algorithms for min-max regret task scheduling problems with interval uncertainty », *in*: 42.*3* (2013) (cit. on p. 59).

[124] Marco Silva, Michael Poss, and Nelson Maculan, « Solution algorithms for minimizing the total tardiness with budgeted processing time uncertainty », *in*: *European Journal of Operational Research* 283.*1* (2020), pp. 70–82, ISSN: 03772217, DOI: 10.1016/j.ejor.2019.10.037 (cit. on pp. 65, 67, 110).

[125] Marcos Melo Silva, Anand Subramanian, Thibaut Vidal, and Luiz Satoru Ochi, « A simple and effective metaheuristic for the Minimum Latency Problem », *in*: *European Journal of Operational Research* 221.*3* (2012), pp. 513–520, ISSN: 0377-2217, DOI: https://doi.org/10.1016/j.ejor.2012.03.044 (cit. on p. 84).

[126] Yu.N. Sotskov and T.-C. Lai, « Minimizing total weighted flow time under uncertainty using dominance and a stability box », *in*: *Computers Operations Research* 39.*6* (2012), Special Issue on Scheduling in Manufacturing Systems, pp. 1271–1289, ISSN: 0305-0548, DOI: https://doi.org/10.1016/j.cor.2011.02.001 (cit. on p. 100).

[127] Yuri N Sotskov and Frank Werner, « the Stability Box in Interval Data for Minimizing the Sum of Weighted Completion Times », *in*: (2011), pp. 14–23, DOI: 10.5220/0003574400140023 (cit. on p. 98).

[128] Allen L Soyster, « Convex programming with set-inclusive constraints and applications to inexact linear programming », *in*: *Operations research* 21.*5* (1973), pp. 1154–1157 (cit. on pp. 31, 61, 62, 77, 101, 102, 139).

[129] Edward F Stafford, « On the development of a mixed-integer linear programming model for the flowshop sequencing problem », *in*: *Journal of the Operational Research Society* 39.*12* (1988), pp. 1163–1174 (cit. on pp. 55, 62, 77, 78, 104).

[130] EF Stafford and FT Tseng, « On 'redundant'constraints in Stafford's MILP model for the flowshop problem », *in*: *Journal of the Operational Research Society* 54.*10* (2003), pp. 1102–1105 (cit. on p. 55).

[131] Wencong Su, Jianhui Wang, and Jaehyung Roh, « Stochastic energy scheduling in microgrids with intermittent renewable energy resources », *in*: *IEEE Transactions on Smart Grid* 5.*4* (2014), pp. 1876–1883, ISSN: 19493053, DOI: 10.1109/TSG.2013.2280645 (cit. on pp. 24, 25).

[132] Eric Taillard, « Benchmarks for basic scheduling problems », *in*: *European Journal of Operational Research* 64.*2* (1993), pp. 278–285 (cit. on p. 85).

[133] Aharon Ben Tal, Boaz Golany, Arcadi Nemirovski, and Jean-Philippe Vial, « Supplier-Retailer Flexible Commitments Contracts: A Robust Optimization Approach », *in*: (2003) (cit. on p. 20).

[134] Fan T Tseng and Edward F Stafford Jr, « Two MILP models for the N× M SDST flowshop sequencing problem », *in*: *International Journal of Production Research* 39.*8* (2001), pp. 1777–1809 (cit. on p. 105).

[135] Fan T Tseng, Edward F Stafford Jr, and Jatinder ND Gupta, « An empirical analysis of integer programming formulations for the permutation flowshop », *in*: *Omega* 32.*4* (2004), pp. 285–293 (cit. on pp. 54, 62, 77, 100).

[136] FT Tseng and EF Stafford, « New MILP models for the permutation flowshop problem », *in*: *Journal of the Operational Research Society* 59.*10* (2008), pp. 1373–1386 (cit. on pp. 53, 100, 102, 104, 106, 118).

[137] Umar Al-Turki, C Fedjki, and Abdulbasit Andijani, « Tabu search for a class of single-machine scheduling problems », *in*: *Computers & Operations Research* 28.*12* (2001), pp. 1223–1230 (cit. on p. 53).

[138] S. Turner and D. Booth, « A new integer programming model for the N job M machine flow shop problem », *in*: *Proceedings of the Midwest Decision Science Institute*, ed. by Schnriederjans MJ, Lincoln: Nebraska, IL, 1986, p. 229 (cit. on pp. 102, 103).

[139] Wim Van Ackooij, Jérôme De Boeck, Boris Detienne, Stefania Pan, and Michael Poss, « Optimizing power generation in the presence of micro-grids », *in*: *European journal of operational research* 271.*2* (2018), pp. 450–461 (cit. on pp. 25, 26).

[140] Karina Vink, Eriko Ankyu, and Michihisa Koyama, « Multiyear microgrid data from a research building in Tsukuba, Japan », *in*: *Scientific data* 6.*1* (2019), pp. 1–9, DOI: 10.1038/sdata.2019.20 (cit. on pp. 21, 40).

[141] Nhat-Vinh Vo and Christophe Lenté, « From maxplus algebra to general lower bounds for the total weighted completion time in flowshop scheduling problems », *in*: *Lecture Notes in Management Science* 6 (2014), pp. 128–137 (cit. on p. 100).

[142] Harvey M Wagner, « An integer linear-programming model for machine scheduling », *in*: *Naval Research Logistics Quarterly* 6.*2* (1959), pp. 131–140 (cit. on pp. 55, 62, 78, 104).

[143] Jishuai Wang and Gang Rong, « Robust optimization model for crude oil scheduling under uncertainty », *in*: *Industrial & Engineering Chemistry Research* 49.*4* (2009), pp. 1737–1748 (cit. on p. 12).

[144] Ran Wang, Ping Wang, and Gaoxi Xiao, « A robust optimization approach for energy generation scheduling in microgrids », *in*: *Energy Conversion and Management* 106.*5613* (Dec. 2015), pp. 597–607, ISSN: 01968904, DOI: 10.1016/j.enconman.2015.09.066 (cit. on p. 25).

[145] Xiao-Yuan Wang, Zhili Zhou, Xi Zhang, Ping Ji, and Ji-Bo Wang, « Several flow shop scheduling problems with truncated position-based learning effect », *in*: *Computers & Operations Research* 40.*12* (2013), pp. 2906–2929, ISSN: 0305-0548, DOI: https://doi.org/10.1016/j.cor.2013.07.001 (cit. on p. 100).

[146] Joseph Warrington, Christian Hohl, Paul J Goulart, and Manfred Morari, « Rolling unit commitment and dispatch with multi-stage recourse policies for heterogeneous devices », *in*: *IEEE Transactions on Power Systems* 31.*1* (2015), pp. 187–197 (cit. on p. 33).

[147] Marco Wiering and Martijn van Otterlo, *Reinforcement Learning: State-of-the-Art*, Springer Publishing Company, Incorporated, 2014, ISBN: 364244685X, 9783642446856 (cit. on p. 133).

[148] Frank Wilcoxon, « Individual Comparisons by Ranking Methods », *in*: *Biometrics Bulletin* 1.*6* (1945), pp. 80–83, ISSN: 00994987, DOI: 10.2307/3001968, URL: http://www.jstor.org/stable/3001968 (cit. on p. 45).

[149] JM Wilson, « Alternative formulations of a flow-shop scheduling problem », *in*: *Journal of the Operational Research Society* 40.*4* (1989), pp. 395–399 (cit. on pp. 55, 62, 64, 77, 78, 102, 123).

[150] Laurence A Wolsey and George L Nemhauser, *Integer and combinatorial optimization*, vol. 55, John Wiley & Sons, 1999 (cit. on p. 30).

[151] Ting Wu et al., « Coordinated energy dispatching in microgrid with wind power generation and plug-in electric vehicles », *in*: *IEEE Transactions on Smart Grid* 4.*3* (2013), pp. 1453–1463, ISSN: 19493053, DOI: 10.1109/TSG.2013.2268870 (cit. on p. 25).

[152] Xiaohua Wu et al., « Stochastic control of smart home energy management with plug-in electric vehicle battery energy storage and photovoltaic array », *in*: *Journal of Power Sources* 333 (2016), pp. 203–212, ISSN: 03787753, DOI: 10.1016/j.jpowsour.2016.09.157 (cit. on pp. 24, 25).

[153] Xiaohua Wu, Xiaosong Hu, Xiaofeng Yin, and Scott J. Moura, « Stochastic Optimal Energy Management of Smart Home With PEV Energy Storage », *in*: *IEEE Transactions on Smart Grid* 9.*3* (2018), pp. 2065–2075, ISSN: 19493053, DOI: 10.1109/TSG.2016.2606442 (cit. on pp. 24, 25).

[154] Takeshi Yamada and Colin R Reeves, « Permutation flowshop scheduling by genetic local search », *in*: (1997) (cit. on p. 53).

[155] Hong-Sen Yan, Qi-Feng Xia, Min-Ru Zhu, Xia-Ling Liu, and Zhi-Min Guo, « Integrated production planning and scheduling on automobile assembly lines », *in*: *Iie Transactions* 35.*8* (2003), pp. 711–725 (cit. on p. 51).

[156] Chaodong Yang, Colin Card, Long X Nghiem, Eugene Fedutenko, et al., « Robust optimization of SAGD operations under geological uncertainties », *in*: *SPE Reservoir Simulation Symposium*, Society of Petroleum Engineers, 2011 (cit. on p. 12).

[157] Shu Hui Yang and Ji Bo Wang, « Minimizing total weighted completion time in a two-machine flow shop scheduling under simple linear deterioration », *in*: *Applied Mathematics and Computation* 217.*9* (2011), pp. 4819–4826, ISSN: 00963003, DOI: 10.1016/j.amc.2010.11.037 (cit. on p. 100).

[158] Elham Yasari, Mahmoud Reza Pishvaie, Farhad Khorasheh, Karim Salahshoor, and Riyaz Kharrat, « Application of multi-criterion robust optimization in water-flooding of oil reservoir », *in*: *Journal of Petroleum Science and Engineering* 109 (2013), pp. 1–11 (cit. on p. 12).

[159] Kuo Ching Ying, « Scheduling the two-machine flowshop to hedge against processing time uncertainty », *in*: *Journal of the Operational Research Society* 66.*9* (2015), pp. 1413–1425, ISSN: 14769360, DOI: 10.1057/jors.2014.100 (cit. on pp. 59–62, 68, 85, 101, 117).

[160] Michael Zachar and Prodromos Daoutidis, « Microgrid/Macrogrid Energy Exchange: A Novel Market Structure and Stochastic Scheduling », *in*: *IEEE Transactions on Smart Grid* 8.*1* (2017), pp. 178–189, ISSN: 19493053, DOI: 10.1109/TSG.2016.2600487 (cit. on pp. 24–26).

[161] Faraj Zarei, Khafiz Muradov, David Davies, et al., « Optimal well work-over scheduling: application of intelligent well control optimisation technology to conventional wells », *in*: *SPE Intelligent Energy Conference & Exhibition*, Society of Petroleum Engineers, 2014 (cit. on p. 97).

[162] Bo Zeng and Long Zhao, « Solving two-stage robust optimization problems using a column-and- constraint generation method », *in*: *Operations Research Letters* 41.*5* (2013), pp. 457–461, ISSN: 01676377, DOI: 10.1016/j.orl.2013.05.003 (cit. on pp. 65, 79, 110).

[163] Yu Zhang, Nikolaos Gatsis, and Georgios B. Giannakis, « Robust distributed energy management for microgrids with renewables », *in*: *IEEE SmartGridComm 2012 November* (2012), pp. 510–515, DOI: 10.1109/SmartGridComm.2012.6486036 (cit. on p. 25).

[164] Yu Zhang, Nikolaos Gatsis, and Georgios B. Giannakis, « Robust energy management for microgrids with high-penetration renewables », *in*: *IEEE Transactions on Sustainable Energy* 4.*4* (2013), pp. 944–953, ISSN: 19493029, DOI: 10.1109/TSTE.2013.2255135 (cit. on pp. 24, 25).