SMART AND SUSTAINABLE FLOW-SHOP SCHEDULING PROBLEMS: SCENARIO-BASED ROBUST OPTIMIZATION AND STRONG HEURISTICS

by

Amir M. FATHOLLAHI-FARD

MANUSCRIPT-BASED THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE IN PARTIAL FULFILLMENT FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, OCTOBER 4TH, 2023

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mrs. Lyne Woodward, Thesis Supervisor
Department of Electrical Engineering, École de technologie supérieure

Mrs. Ouassima Akhrif, Thesis Co-Supervisor
Department of Electrical Engineering, École de technologie supérieure

Mr. Marc Paquet, President of the Board of Examiners
Department of Systems Engineering, École de technologie supérieure

Mr. Amin Chaabane, Member of the Jury
Department of Systems Engineering at École de technologie supérieure

Mr. Leandro C. Coelho, External Member of the Jury
Operations and Decision Systems Department, Université Laval

THIS THESIS WAS PRENSENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND PUBLIC

ON “SEPTEMBER” “15”, “2023”

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to Prof. Lyne Woodward, my supervisor, and Prof. Ouassima Akhrif, my co-supervisor, for their invaluable support throughout the completion of my dissertation. Their guidance, availability, encouragement, and professionalism have significantly enhanced my project journey and positively influenced my approach to supervising others.

I would also like to extend my sincere thanks to the members of the jury for their gracious agreement to serve as examiners for this thesis. I am truly honored by their presence and grateful for their constructive feedback and comments, which have greatly contributed to the improvement and clarity of the study.

Furthermore, I wish to acknowledge the financial support provided by the discovery grant program from the National Sciences and Engineering Research Council of Canada (NSERC), grant number RGPIN-2019–05853. This support has been instrumental in the successful completion of my research.

Once again, I express my heartfelt appreciation to Prof. Lyne Woodward, Prof. Ouassima Akhrif, the jury members, and NSERC for their significant contributions to my academic journey and the accomplishment of my thesis.

Problèmes d'ordonnancement des magasins de flux intelligents et durables: scenario optimisation robuste et heuristique forte

Amir M. FATHOLLAHI-FARD

RÉSUMÉ

Cette thèse de doctorat est consacrée au développement d'une approche intelligente et durable au problème d'ordonnancement de permutation d'ateliers distribués (DPFSP) en utilisant des modèles d'optimisation pratiques, des reformulations efficaces, des heuristiques et des métaheuristiques avancées. Le DPFSP est une extension du problème d'ordonnancement de permutation d'ateliers (PFSP) et sert de base au problème proposé. La principale distinction entre le DPFSP et le PFSP réside dans leurs domaines d'ordonnancement respectifs. Alors que le PFSP se concentre sur l'ordonnancement des tâches au sein d'une seule usine, le DPFSP relève le défi plus complexe de l'ordonnancement des tâches distribuées dans plusieurs usines. Bien que des recherches antérieures aient contribué au domaine du DPFSP, ce projet de doctorat se distingue en intégrant les concepts de durabilité, d'ordonnancement en temps réel et d'optimisation robuste dans le cadre du DPFSP. L'objectif principal de cette recherche est d'intégrer des critères environnementaux et sociaux basés sur le Triple Bilan (TBL) afin de respecter les lignes directrices des objectifs de développement durable. En tenant compte de critères tels que la consommation d'énergie, les opportunités d'emploi et les jours de travail perdus, un modèle d'optimisation multi-objectif et un algorithme métaheuristique multi-objectif efficace sont développés.

Une autre lacune majeure dans le domaine de l'ordonnancement de la production concerne la collecte intelligente, l'analyse et la conversion des données en informations exploitables dans des stratégies de prise de décision en temps réel pour les systèmes de production. En réponse à ce grand défi, le deuxième objectif de ce projet de doctorat est de traiter l'incertitude dans le DPFSP en le modélisant dans le cadre d'optimisation en temps réel de l'industrie 4.0. Une approche d'optimisation en temps réel est proposée pour gérer la réattribution des tâches aux machines en tenant compte des temps de traitement incertains, de l'arrivée de nouvelles tâches ou des pannes de machines. En incorporant les concepts de l'industrie 4.0, un modèle

d'optimisation complet utilisant différents modes de production manuels et automatisés est proposé, et diverses stratégies et politiques d'ordonnancement en temps réel sont examinées dans ce modèle. Pour le résoudre, des heuristiques constructives, des relaxations lagrangiennes et des reformulations de décomposition de Benders sont étudiées.

Alors que le deuxième objectif aborde l'incertitude dans une certaine mesure, le troisième objectif utilise une approche d'optimisation robuste basée sur des scénarios pour traiter efficacement l'incertitude dans le DPFSP en considérant tous les scénarios possibles. L'objectif final de ce projet de doctorat est de relever les défis du DPFSP intelligent et durable grâce au développement d'un cadre d'optimisation complet. Ce cadre combine un modèle d'optimisation robuste basé sur des scénarios et un algorithme métaheuristique avancé basé sur la recherche locale adaptative du voisinage (ALNS) en utilisant diverses heuristiques et algorithmes de recherche locale. En utilisant une approche d'optimisation robuste basée sur des scénarios, le cadre examine toute une gamme de scénarios possibles pouvant survenir en raison de diverses perturbations dans les horaires de production. Ces perturbations peuvent inclure des pannes de machines, l'arrivée de nouvelles tâches ou des variations des temps de traitement des tâches. En incorporant ces incertitudes dans le processus d'optimisation, le cadre permet d'identifier des horaires robustes et résilients aux circonstances imprévues.

Dans l'ensemble, ce projet de doctorat représente une avancée significative dans le domaine du DPFSP en tirant parti des principes de durabilité, d'ordonnancement en temps réel et d'optimisation robuste. Grâce à l'application de modèles d'optimisation pratiques, de reformulations efficaces, d'heuristiques et de métaheuristiques, cette recherche vise à relever les défis uniques posés par l'ordonnancement des tâches dans des usines distribuées. En contribuant au développement de systèmes de production plus intelligents et durables, ce travail a des implications considérables pour le domaine et l'industrie dans son ensemble.

Mots-clés: Production intelligente; Production durable; Ateliers de permutation distribués; Métaheuristiques;

Smart and sustainable flow-shop scheduling problems: Scenario-based robust optimization and strong heuristics

Amir M. FATHOLLAHI-FARD

ABSTRACT

This Ph.D. thesis is dedicated to the development of a smart and sustainable approach to the Distributed Permutation Flow Shop Scheduling Problem (DPFSP) through the utilization of practical optimization models, efficient reformulations, heuristics, and advanced metaheuristics. The DPFSP is an extension of the Permutation Flow Shop Scheduling Problem (PFSP) and serves as its foundational model. The key distinction between the DPFSP and the PFSP lies in their respective scheduling scopes. While the PFSP focuses on scheduling tasks within a single plant, the DPFSP addresses the more complex challenge of scheduling tasks across multiple distributed factories.

While prior research has made contributions to the field of DPFSP, this Ph.D. project stands out by incorporating the concepts of sustainability, real-time scheduling, and scenario-based robust optimization into the DPFSP framework. The primary objective of this research is to integrate environmental and social criteria based on the Triple Bottom Line (TBL) to meet the guidelines of the Sustainable Development Goals (SDGs). By considering criteria such as energy consumption, job opportunities, and lost workdays, a multi-objective optimization model and an efficient multi-objective metaheuristic algorithm are developed.

Another critical research gap in the field of production scheduling involves the intelligent collection, analysis, and conversion of data into actionable information using real-time decision-making strategies for production systems. In response to this grand challenge, the second objective of this Ph.D. project is to address the uncertainty in the DPFSP by modeling it within the real-time optimization framework of Industry 4.0. A real-time optimization approach is proposed to handle task reassignment to machines under uncertain process times, new task arrivals, or planned machine breakdowns. By incorporating the concepts of Industry 4.0, a comprehensive optimization model using different manual and automated modes of production is proposed and various real-time scheduling strategies and policies

are examined into this model. For solving it, constructive heuristics, Lagrangian relaxation and Benders decomposition reformulations are studied.

While the second objective addresses uncertainty to some extent, the third objective utilizes a scenario-based robust optimization approach to efficiently address uncertainty in the DPFSP by considering all possible scenarios. The final objective of this Ph.D. project is to address the challenges of the smart and sustainable DPFSP through the development of a comprehensive optimization framework. This framework combines a scenario-based robust optimization model and an advanced metaheuristic algorithm based on adaptive large neighborhood search (ALNS) using various heuristic and local search algorithms. By employing a scenario-based robust optimization approach, the framework considers a range of possible scenarios that may arise due to various disruptions in production schedules. These disruptions can include machine breakdowns, arrival of new tasks, or variations in task processing times. By incorporating these uncertainties into the optimization process, the framework enables the identification of schedules that are robust and resilient to unforeseen circumstances.

Overall, this Ph.D. project represents a significant advancement in the field of DPFSP by leveraging the principles of sustainability, real-time scheduling, and robust optimization. Through the application of practical optimization models, efficient reformulations, heuristics, and metaheuristics, this research aims to address the unique challenges posed by scheduling tasks across distributed factories. By contributing to the development of smarter and more sustainable production systems, this work has far-reaching implications for the field and industry as a whole.

Keywords: Smart Production, Sustainable Production, Distributed Permutation Flow-Shops; Metaheuristics;

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ABREVIATIONS

PFSP	Permutation flow-shop scheduling problem
DPFSP	Distributed permutation flow-shop scheduling problem
TBL	Triple bottom line
SDG	Sustainable development goals
AI	Artificial Intelligence
Industry 4.0	The fourth industrial revolution
IoT	Internet of things
SEO	Social engineering optimizer
KCA	Knowledge-based cooperative algorithm
ALNS	Adaptive large neighborhood search
SA	Simulated annealing
MOBSO	Multi-objective brain storm optimization
MOMOA	Multi-objective memetic optimization algorithm
ML	Machine Learning
DEMO	Discrete evolutionary multi-objective optimization
BFS	Blocking flow-shop scheduling

MOWOA	Multi-objective whale optimization algorithm
LSEO	Learning-based social engineering optimizer
DSS	Decision Support System
FJS	Flexible job-shop scheduling
JSP	Job-shop scheduling problem
RL	Reinforcement learning
HPSMO	Hybrid Pareto spider monkey optimization
SGA	Scenario-based genetic algorithm
MBRIG	Multi-objective biased randomized iterated greedy;
GDP	Gross Domestic Product
CWP	Canadian Wood Products

LIST OF MEASUREMENT UNITS

s	Second (time unit)
m	Minutes (time unit)
h	Hours (time unit)
kWh	Kilowatt hour (energy unit)
BTU	British Thermal Unit (energy unit)

INTRODUCTION

In today's complex and dynamic manufacturing environments, efficient production scheduling plays a vital role in optimizing resources, reducing costs, and enhancing overall productivity. The Distributed Permutation Flow Shop Scheduling Problem (DPFSP), which extends the well-known Permutation Flow Shop Scheduling Problem (PFSP), presents a challenging scenario where tasks need to be scheduled across multiple distributed factories. This Ph.D. thesis is dedicated to the development of a smart and sustainable approach to address the DPFSP by integrating practical optimization models, efficient reformulations, heuristics, and advanced metaheuristics. By considering the principles of sustainability, real-time scheduling, and scenario-based robust optimization, this research aims to revolutionize the field of production scheduling.

The motivation behind this research stems from the growing need for sustainable manufacturing practices and the increasing complexity of scheduling tasks across distributed factories. Traditional scheduling approaches often overlook crucial factors such as environmental impact and social considerations. As industries strive to align with the Sustainable Development Goals (SDGs) and the Triple Bottom Line (TBL), it becomes imperative to develop scheduling models that integrate environmental, social, and economic criteria (Fathollahi-Fard et al., 2020). Furthermore, with the emergence of Industry 4.0 and its emphasis on real-time decision-making, there is a pressing need to address uncertainty and disruptions in production systems effectively.

The sustainable DPFSP models developed in this thesis offer versatile applicability across a range of industries, including but not limited to sectors such as automobile manufacturing and wood production that commonly employ DPFSP practices. Chapter 3 of this thesis provides a specific illustration of our model's application within the context of wood production. We focus on Canadian Wood Products (CWP), a prominent leader in the Canadian wood industry, known for its distinguished role as a premier producer and distributor of wood products throughout North America. CWP manages an extensive product portfolio, spanning softwoods, industrial lumbers, and architectural lumbers. To optimize their production processes effectively, CWP employs specialized operational modes on their production machinery, demanding precise scheduling. This intricate production cycle encompasses six essential tasks: cutting, custom processing, drying, classifying, storing, and loading. CWP's operational landscape extends to three pivotal production and distribution centers strategically located in Buffalo, Montreal, and Concord. Consequently, each of these facilities presents a flow shop scheduling scenario, making it a fitting application for our thesis. By utilizing CWP as a real-world benchmark, we aim to provide tangible evidence of the successful implementation of our optimization methodologies and sustainability-focused approach in navigating the complexities of practical production scenarios. This application yields valuable insights and practical solutions for the adoption of sustainable scheduling practices, particularly within the wood industry.

Generally, this Ph.D. project makes substantial contributions to the field of DPFSP by addressing the unique challenges of scheduling tasks across distributed factories while incorporating the principles of sustainability, real-time scheduling, and scenario-based robust optimization. It also showcases an experimental setup used to validate the proposed models and algorithms while analyzing different versions of DPFSP. The key contributions can be summarized as follows:

Smart and Sustainable DPFSP: This research integrates environmental and social criteria based on the TBL to develop a multi-objective optimization model for the DPFSP. By considering sustainability aspects such as energy consumption, job opportunities, and lost workdays, the developed model contributes to the alignment of production systems with the SDGs. Additionally, an efficient multi-objective metaheuristic algorithm using the social engineering optimizer and adaptive search learning is proposed to solve the multi-objective optimization problem effectively.

The focus on sustainability in the DPFSP is motivated by the pressing need for environmentally responsible manufacturing practices. Traditional scheduling approaches often neglect the environmental impact of production processes. By incorporating sustainability criteria into the optimization model, such as reducing energy consumption, the research aims to contribute to the development of greener and more sustainable production systems. In addition, the social aspect of sustainability is also a critical consideration. The DPFSP provides an opportunity to evaluate job opportunities and lost workdays in the scheduling process. By optimizing the allocation of tasks across distributed factories, the research aims to promote fair employment and minimize lost workdays, thereby contributing to the social well-being and stability of the workforce.

Real-Time Scheduling in DPFSP: The project addresses the uncertainty in the DPFSP by modeling it within the real-time optimization framework of Industry 4.0. A comprehensive optimization model is proposed, incorporating different manual and automated modes of production, along with various real-time scheduling strategies and policies. This framework enables intelligent decision-making for task reassignment while considering unexpected events such as new task arrivals and machine breakdowns, thereby enhancing the adaptability and responsiveness of production systems.

Industry 4.0 emphasizes real-time data collection and analysis, enabling proactive decision-making for optimal production scheduling. By leveraging the capabilities of Industry 4.0 technologies, such as the Internet of Things (IoT), cyber-physical systems, and real-time data analytics, the research aims to develop scheduling strategies that can dynamically adapt to changing conditions and uncertainties. To this end, we develop a real-time scheduling approach for the DPFSP to address real-time conditions and uncertainties.

Real-time scheduling encompasses a set of strategies and policies designed to handle uncertainties, such as machine breakdowns or the arrival of new tasks. Rescheduling, as a core element, ensures that production schedules remain optimal even when disruptions occur. The rescheduling policy dictates when and how the rescheduling should be made. By reallocating tasks to available machines or adapting the production sequence, our proposed real-time scheduling approach acknowledges the dynamic nature of modern manufacturing environments. This capability minimizes the impact of unexpected events, ensuring sustained high levels of productivity and efficiency.

Scenario-Based Robust Optimization: Uncertainty in the DPFSP is efficiently addressed through a scenario-based robust optimization approach. By considering a range of possible disruptions, such as machine breakdowns, new task arrivals, and variations in task processing times, the developed framework identifies schedules that are robust and resilient to unforeseen circumstances. This contributes to the development of reliable and flexible production schedules that can withstand unexpected events.

The scenario-based robust optimization approach recognizes that uncertainties are inherent in manufacturing processes. By considering multiple scenarios and their corresponding probabilities, the research aims to develop schedules that perform well across a range of possible disruptions. This approach provides decision-makers with insights into the robustness of different schedules and enables them to make informed decisions that minimize the negative impact of uncertainties. The incorporation of variations in task processing times acknowledges the inherent variability in real-world production environments. By considering different possible processing times for tasks, the research aims to develop schedules that are robust to variations, ensuring that production processes remain efficient even when faced with fluctuations in task durations.

Optimization Framework using an advanced Metaheuristic Algorithm: A comprehensive optimization framework is developed by combining a scenario-based robust optimization model and an advanced metaheuristic algorithm based on adaptive large neighborhood search (ALNS). This framework leverages various heuristic and local search algorithms to provide efficient and effective solutions for the DPFSP. The optimization framework combines the strengths of robust optimization and metaheuristics to tackle the DPFSP effectively. The scenario-based robust optimization model provides insights into the performance of different schedules under various disruptive scenarios, allowing decision-makers to choose schedules that offer the best trade-off between performance and robustness.

This advanced metaheuristic algorithm based on ALNS offers a powerful search strategy for exploring the solution space and finding high-quality schedules. By incorporating adaptive mechanisms and neighborhood structures, the algorithm can dynamically adapt its search process to different instances of the DPFSP, leading to efficient and effective solutions.

The utilization of various heuristic and local search algorithms within the ALNS framework ensures a comprehensive exploration of the solution space, increasing the chances of finding optimal or near-optimal schedules. These algorithms leverage domain-specific knowledge and problem characteristics to guide the search process, improving the efficiency and effectiveness of the optimization framework.

In conclusion, this Ph.D. project represents a significant advancement in the field of DPFSP by leveraging the principles of sustainability, real-time scheduling, and scenario-based robust optimization. Through the application of practical optimization models, efficient reformulations, heuristics, and metaheuristics, this research aims to address the unique challenges posed by scheduling tasks across distributed factories. By contributing to the development of smarter and more sustainable production systems, this work has far-reaching implications for the field and industry as a whole.

The following thesis is structured into six comprehensive chapters, each addressing specific aspects of the research topic. Chapter 1 sets the foundation for the thesis by outlining the objectives and research methodology employed throughout the study. It clearly defines the research questions and goals that this Ph.D. project aims to address.

Chapter 2 presents an extensive review of the literature relevant to the research topic. It explores the existing studies, publications, and approaches related to the DPFSP and its extensions, as well as sustainability considerations and real-time optimization in production scheduling. It provides a comprehensive overview of the state-of-the-art techniques, optimization models, and metaheuristics employed in solving scheduling problems. By examining the existing literature, this chapter offers insights into the evolution of the field and establishes the theoretical background for the subsequent chapters.

Chapter 3 focuses on the first paper extracted during this Ph.D. thesis which was published in the Journal of Industrial Information Integration. The chapter presents a multi-objective optimization model for the DPFSP based on different operating modes of machines and sustainability criteria including energy consumption, job opportunities, and lost working days. The feasibility of the model is validated through a numerical example from the wood industry. An efficient multi-objective metaheuristic algorithm using the social engineering theory and an adaptive learning strategy is developed to solve this multi-objective optimization problem. To validate the results of our multi-objective metaheuristic algorithm, the epsilon constraint method is utilized. This makes it possible to examine the optimality of the Pareto solutions. Our approach is then compared with state-of-the-art multi-objective metaheuristic methods. This chapter provides a detailed discussion of the results, conclusions, and implications of the paper's findings for the sustainable production in the wood industry.

In Chapter 4, the second paper extracted during this Ph.D. thesis is presented. A short version of this chapter was submitted to the International Conference on Industry 4.0 and Smart Manufacturing (ISM 2023). The full version is under review in the Journal of Industrial Information Integration. In this chapter, we develop a sustainable DPFSP using different real-time scheduling strategies and policies. The chapter discusses innovative approaches utilized to address uncertainties and disruptions in the DPFSP within the framework of Industry 4.0. It delves into efficient constructive heuristic algorithms and reformulation techniques based on Lagrangian relaxation and Benders decomposition. All the heuristics and reformulations are validated against the exact solver and compared with each other. Generally, this chapter highlights the unique contributions of the paper to the field of smart and sustainable DPFSP and discusses the practical implications of its findings.

Chapter 5 focuses on the third paper derived from this Ph.D. thesis which is under review in Annals of Operations Research. It provides a problem formulation of a scenario-based robust optimization approach for the sustainable DPFSP, and an advanced metaheuristic algorithm based on the ALNS using different removal and construction heuristics as well as a local search algorithm. The chapter presents the experimental setup, performance evaluation, and comparison with existing approaches. Addressing the challenges of the smart and sustainable DPFSP through the development of a comprehensive optimization framework, this chapter ends with a discussion of the results and potential applications of the paper's findings.

The final chapter i.e., Chapter 6 presents a comprehensive summary of the research conducted throughout the Ph.D. project. It revisits the research objectives and provides a detailed analysis of the contributions made by this research. The chapter highlights the significance of integrating sustainability and different uncertainty approaches in addressing the DPFSP and discusses the practical implications of the findings. It also reflects on the limitations and potential areas for future research. In conclusion, this chapter provides a comprehensive synthesis of the research outcomes and their broader implications.

CHAPTER 1

RESEARCH QUESTIONS, OBJECTIVES AND RESEARCH METHODOLOGY

1.1 Research questions

The research questions constituting the backbone of this doctoral thesis on the smart and sustainable approach to the DPFSP are stated as follows:

- How can the principles of sustainability be integrated into the optimization models and methodologies for the DPFSP?
- How can real-time scheduling strategies and policies be effectively employed in the DPFSP to handle uncertainties and disruptions?
- How can scenario-based robust optimization techniques be employed to improve the robustness and resilience of the DPFSP schedules?
- How can advanced metaheuristics be leveraged to optimize the DPFSP schedules?

Generally, these research questions collectively drive the investigation into developing a smart and sustainable approach to the DPFSP. By addressing these questions, the research aims to contribute to the field by integrating sustainability principles, real-time scheduling strategies, and scenario-based robust optimization techniques into the scheduling process, ultimately enhancing the efficiency, resilience, and sustainability of production systems.

1.2 Research scope and limitations

While this thesis contributes valuable insights to the field of smart and sustainable flow-shop scheduling problems, it also has certain limitations that should be acknowledged:

- **Generalizability**: The research findings and proposed methodologies are specific to the DPFSP and may not be directly applicable to other scheduling problems or industries. The scope of the research is limited to the context of distributed permutation flow shop environments.
- **Simplifying assumptions**: To develop practical optimization models and methodologies, certain assumptions and simplifications have been made based on the principles of DPFSP. These assumptions may not fully capture the complexities and nuances of real-world production systems.
- **Data availability**: The effectiveness of the proposed approaches heavily relies on the availability and quality of data. Limited access to real-time data or incomplete data may impact the accuracy and performance of the developed models and algorithms.
- **Scalability**: The scalability of the proposed methodologies may pose a challenge when dealing with large-scale production systems. As the size and complexity of the problem increase, the computational requirements and the practicality of the proposed approaches may need to be carefully considered.
- **Technological constraints**: The implementation of Industry 4.0 technologies and real-time optimization strategies may require specific infrastructure, resources, and technological capabilities. The feasibility and adoption of these technologies may vary across different manufacturing settings.

By acknowledging these limitations, the thesis ensures a clear understanding of the research boundaries while providing valuable contributions to the field of smart and sustainable flow-shop scheduling problems.

1.3 Research contributions

The research contributions of this thesis are threefold based on the three papers as follows: The first paper addresses the concept of sustainable development in the context of production scheduling by conceptualizing an energy-efficient Distributed DPFSP into a sustainable DPFSP. The study introduces a novel multi-objective mixed integer linear model that aims to minimize total energy consumption while simultaneously maximizing social factors such as job opportunities and reducing lost working days. A multi-objective learning-based heuristic, an extension of the Social Engineering Optimizer (SEO), is proposed to handle the complexity of the model. The paper showcases the applicability of the model in the wood industry context in Canada through simulated tests, comparisons with other methods, and sensitivity analyses. The research contribution lies in the integration of sustainability criteria into the DPFSP and the development of a novel optimization model and heuristic.

The second paper contributes to both real-time scheduling and sustainable production fields by redefining the sustainable distributed permutation flow-shop scheduling problem. The proposed model aims to minimize the makespan, reduce energy consumption, and the number of lost working days, while simultaneously increasing job opportunities. Real-time scheduling is performed using predictive-reactive and proactive-reactive strategies, with two rescheduling policies considered: continuous and event-driven. Various reformulations and heuristics are defined to address the complexity of the model. The paper provides a comprehensive analysis of the results, highlighting the advantages of the predictive-reactive scheduling strategy and the event-driven rescheduling policy. The research contribution lies in redefining the scheduling problem to incorporate sustainability criteria and providing practical insights for production managers on real-time scheduling in dynamic environments.

Finally, the third paper focuses on sustainable production scheduling by formulating the Sustainable Distributed Permutation Flow-shop Scheduling Problem (SDPFSP). The study considers economic, environmental, and social criteria and incorporates multiple uncertainties such as machine breakdowns, processing time, and random job arrivals. A scenario-based robust optimization model is proposed to minimize the expected makespan and its deviations from probabilistic scenarios. The paper introduces the ALNS algorithm as a novel metaheuristic approach to tackle this complex optimization problem. The algorithm employs constructive heuristics, removal and construction heuristics, and local search to explore the search space efficiently. Computational studies, comparisons with exact solvers and state-of-the-art metaheuristics, and sensitivity analyses are conducted to validate and evaluate the proposed model and algorithm. The research contribution lies in formulating the SDPFSP with sustainability considerations and introducing a new metaheuristic algorithm for solving it.

Overall, the contributions of this thesis are multi-faceted. It advances the field of production scheduling by incorporating principles of sustainability, real-time scheduling, and robust optimization into the DPFSP. The research introduces novel multi-objective optimization models, efficient reformulations, and heuristics for addressing the DPFSP with sustainability criteria, such as energy consumption, job opportunities, and lost working days. It explores real-time decision-making strategies and rescheduling policies to enhance adaptability and responsiveness in dynamic production environments. Furthermore, the thesis tackles uncertainties in production scheduling through scenario-based robust optimization approaches, considering multiple disruptions and variations. The development of the ALNS algorithm provides an innovative metaheuristic solution to address the complexity of the scheduling problem. Collectively, these contributions offer practical implications for production managers and pave the way for more efficient and sustainable production systems.

1.4 Objectives

Although numerous studies have been conducted on the DPFSP, a comprehensive framework for establishing a smart and sustainable DPFSP has yet to be developed. Thus, the primary objectives of this Ph.D. project are as follows:

- **Addressing a sustainable DPFSP using a deterministic model:** A multi-objective optimization model is proposed to integrate economic, environmental, and social goals into the DPFSP, considering different operating modes ranging from manual to automated. The model aims to minimize the makespan, and energy consumption, and maximize social benefits, including job opportunities while reducing lost working days. To tackle the complexity of the model, efficient exact and metaheuristic algorithms are developed.

- **Addressing a smart and sustainable DPFSP using real-time scheduling:** The second optimization model focuses on minimizing the makespan while optimizing energy consumption and the number of lost working days, with the constraint of creating job opportunities within allowable bounds. This model incorporates uncertainties such as machine breakdowns, variable processing time, and new task arrivals. Two rescheduling policies (continuous and event-driven) and two scheduling strategies (predictive-reactive and proactive-reactive) are employed to handle these uncertainties. Efficient reformulations using Lagrangian relaxation and Benders decomposition, as well as four constructive heuristics, are proposed to solve this optimization model.
- **Addressing the smart and sustainable DPFSP using a robust optimization model:** The third research objective focuses on addressing the uncertainty inherent in the smart and sustainable DPFSP through a scenario-based robust optimization approach. This approach considers uncertain factors such as new task arrivals, variable processing time, and machine breakdowns. Probabilistic functions and robust optimization theories are applied to model and manage these uncertainties. To solve this robust optimization model, an advanced metaheuristic algorithm named Adaptive Large Neighborhood Search (ALNS) is utilized. The ALNS algorithm incorporates different removal, construction, and local search heuristics to efficiently explore the search space and identify robust solutions.

1.5 Research methodology

The research methodology of this thesis adopts a systematic and structured approach to address the research objectives and contribute to the field of smart and sustainable flow-shop scheduling problems. The methodology encompasses several key components, including problem definition, model development, algorithm design, and experimental validation as explained in Figure 1.1.

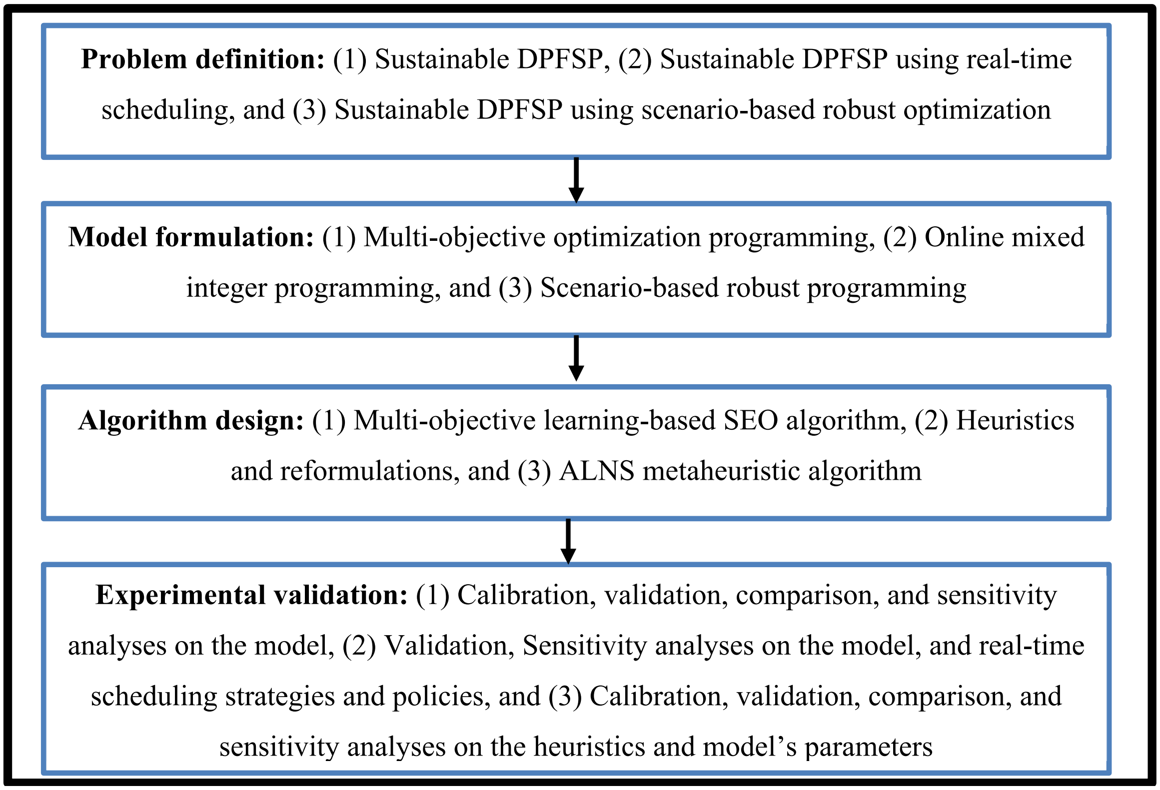


Figure 1.1 Research methodology in this thesis

To begin with, the problem definition stage involves a comprehensive review of the existing literature and identification of the gaps and challenges in the field of DPFSP. This process helps in clearly defining the research scope, objectives, and research questions that guide the subsequent stages of the study.

Next, the model development phase focuses on designing mathematical optimization models that capture the complexities of the DPFSP and incorporate sustainability and real-time scheduling aspects. Different aspects such as economic, environmental, and social criteria, operating modes, and uncertainties are considered in the formulation of the models. The development of the models involves defining the decision variables, objective functions, and constraints that represent the problem accurately.

Following the model development, the algorithm design stage entails the design and implementation of efficient algorithms and metaheuristics to solve the formulated optimization models. This phase also includes the investigation of efficient reformulations such as Lagrangian relaxation and Benders decomposition to enhance the computational efficiency of the models. Various algorithmic techniques such as exact methods, metaheuristics, and heuristics are explored and tailored to the specific requirements of the problem. Special attention is given to the development of advanced metaheuristic algorithms, including the ALNS, which can effectively handle the complexity and uncertainty of the DPFSP.

To validate the effectiveness and practicality of the proposed approaches, an extensive experimental validation phase is conducted. This involves the use of numerical case studies and real-world production scenarios to test the developed models and algorithms. Simulations and computational experiments are performed to analyze the performance of the proposed methodologies in terms of solution quality, computational efficiency, and robustness in dealing with uncertainties. The results are compared with existing state-of-the-art methods and evaluated based on various performance metrics.

Overall, the research methodology of this thesis encompasses problem formulation, model development, algorithm design, experimental validation, and analysis, all of which are executed systematically and rigorously. By following this methodology, the thesis aims to provide novel and practical solutions to the challenges of the DPFSP, contributing to the advancement of the field and promoting the development of smarter and more sustainable production systems.

CHAPTER 2

CRITICAL REVIEW OF THE LITERATURE

The research area of production scheduling is highly active, with numerous optimization algorithms developed to tackle the complexity of this problem (Graves, 1981; Tang et al., 2001; Gahm et al., 2016). As a result, several review papers have been published, focusing on different optimization models and algorithms in the literature (Gao & Chen, 2011; Lin et al., 2013; Naderi & Ruiz, 2014; Shao et al., 2020). In this context, this study aims to highlight the key contributions that support the proposed integration of Industry 4.0 concepts into the sustainable distributed flow-shop scheduling problem. To achieve this, the literature review is structured into five subsections.

Firstly, the review introduces the fundamental models that are relevant to the research area. These models serve as a foundation for understanding the subsequent discussions. Following that, the focus shifts towards exploring the models specifically defined and utilized in the context of Industry 4.0. This subsection provides insights into the advancements made in integrating Industry 4.0 principles into production scheduling.

Additionally, the research area of sustainable and energy-efficient production scheduling is thoroughly examined. This analysis highlights the studies that have addressed sustainability and energy considerations in the scheduling domain, providing a comprehensive overview of this important aspect.

Furthermore, all relevant studies in the field of DPFSP are reviewed. This subsection delves into the existing literature to identify the key findings, methodologies, and limitations of previous research efforts. Finally, by comparing the main contributions of this Ph.D. project with the existing literature, the review identifies research gaps. These gaps represent opportunities for further investigation and improvement in the field of sustainable distributed flow-shop scheduling, incorporating Industry 4.0 principles.

2.1 Base models

To illustrate the distinctions among various production scheduling models, this study introduces different fundamental base models for the production scheduling. The first primary model is the Job-shop Scheduling Problem (JSP) (Manne (1960), Błażewicz et al., (1996), Xiong et al., (2022)). The JSP includes *n* jobs which are a combination of many operations to be processed on *M* machines during a planning horizon. Each machine performs one operation at a time and this operation must be completed on that machine without interruption. Assume that $j \in A_t = \{0, 1, \ldots, n, n+1\}$ denotes the set of operations where operations 0 and $n+1$ represent the initial and final operations, respectively, and have no duration. Moreover, the set of machines is defined by $m \in \mathcal{M} = \{1, 2, \ldots, M\}$. In the JSP, all jobs are available at time zero to be processed on the set of machines. The goal of JSP is to find an optimal sequence of jobs to minimize the makespan (C_{max}) which is the completion time of the last operation in the schedule.

The JSP has two main constraints. The first one is the precedence constraint to force each operation *j* to be scheduled after all predecessor operations. Assume that p_j and c_j represent respectively the process time and the completion time of operation *j*. The process time is the time needed to process the operation. The completion time is the moment at which this operation is completed. Assume also that A_t is the set of operations being processed at time *t*. The main decision variable is a binary variable r_{jm} which is 1 if the operation *j* requires machine *m* to be processed and 0 otherwise. With regards to this definition, the conceptual model for the JSP is formulated as follows:

$$Z = min(C_{max}) \tag{2.1}$$

s.t.

$$c_{j-1} \leq c_j - p_j \times r_{jm} \qquad \forall\, j \in A_t, m \in \mathcal{M}, \tag{2.2}$$

$$C_{max} \geq c_{n+1} \qquad \forall\, n \in \mathcal{N} \tag{2.3}$$

$$\sum_{j \in A_t} r_{jm} \leq 1 \qquad \forall\, m \in \mathcal{M}, t \geq 0 \tag{2.4}$$

$$c_j, C_{max} \geq 0, \qquad \forall\, m \in \mathcal{M},\ j \in A_t \tag{2.5}$$

$$r_{jm} \in \{0,1\}, \qquad \forall\, m \in \mathcal{M}, j \in A_t \tag{2.6}$$

The objective function (2.1) minimizes the makespan. Constraints (2.2) and (2.3) are the precedence relations between operations. Constraint set (2.4) confirms that only one machine is able to process the operation j at a time. Decision variables are defined in (2.5) and (2.6).

Another base model is the Flow-shop Scheduling Problem (FSP) modeled by Wilson (1989) and Gonçalves et al., (2005). In this problem, there are N tasks ($n \in \mathcal{N}$) and M machines ($m \in \mathcal{M}$). Each task has one position ($i \in \mathcal{N}$) on each machine. The operation of task n on machine m (O_{nm}) has the process time (P_{nm}). To highlight the difference between JSP and FSP, the number of manufactured products in the FSP, is higher than in the job-shop environment. However, the variety of products for the JSP is generally higher than for the flow-shop environment.

In a Permutation FSP (PFSP), although the sequence of machines is fixed, the solution of the PFSP consists of the optimal sequence of all tasks on all machines (Gonçalves et al., 2005; Ruiz, & Stützle, 2007). The main decision variable (X_{nim}) is the assignment of task n which is set as position i on machine m as a binary variable. Other variables are the starting time for position i on machine m (ST_{im}), completion time for position i on machine m (C_{im}) and the makespan (C_{max}).

Based on the aforementioned definition, the PFSP is formulated as follows:

$$Z = \min(C_{max}) \tag{2.7}$$

s.t.

$$\sum_{i=1}^{N} X_{nim} = 1, \qquad \forall n \in \mathcal{N}, m \in \mathcal{M} \tag{2.8}$$

$$\sum_{n=1}^{N} X_{nim} = 1, \qquad \forall i \in \mathcal{N}, m \in \mathcal{M} \tag{2.9}$$

$$C_{im} = ST_{im} + \sum_{n=1}^{N} (X_{nim} \times P_{nm}), \qquad \forall i \in \mathcal{N}, m \in \mathcal{M} \tag{2.10}$$

$$C_{im} \geq ST_{i,m-1} + \sum_{n=1}^{N} (X_{nim} \times P_{nm}), \qquad \forall i \in \mathcal{N}, m > 1, m \in \mathcal{M} \tag{2.11}$$

$$C_{im} \geq ST_{i-1,m} + \sum_{n=1}^{N} (X_{nim} \times P_{nm}), \qquad \forall i > 1, i \in \mathcal{N}, m \in \mathcal{M} \tag{2.12}$$

$$C_{max} \geq CT_{im}, \qquad \forall i \in \mathcal{N}, m \in \mathcal{M} \tag{2.13}$$

$$ST_{im}, C_{im}, C_{max} \geq 0, \qquad \forall i \in \mathcal{N}, m \in \mathcal{M} \tag{2.14}$$

$$X_{nim} \in \{1,0\}, \qquad \forall i, n \in \mathcal{N}, m \in \mathcal{M} \tag{2.15}$$

The objective function given in (2.7) aims to minimize the makespan. Constraints (2.8) and (2.9) represent the need for each task to be assigned to a unique position. Constraints (2.10) to (2.12) are the precedence relations between each position and each machine to compute the completion time. The constraint set (2.13) calculates the makespan from all completion times for all positions and machines. Finally, the continuous variables are defined in (2.14) and the main binary variable is defined in (2.15).

Finally, Naderi & Ruiz (2010) proposed a Distributed PFSP (DPFSP) for the first time in which the PFSP is distributed by different factories. Similar to the PFSP, we have *M* machines and *N* tasks and for each task, there is an *i* position. There are *F* factories to which a number (A_f) of tasks must be assigned. The main decision variable (X_{nimf}) is the assignment of task *n* which is set as position *i* on machine *m* at the factory *f* as a binary variable. The operation of a task n on a machine m in a factory f is defined by O_{nmf}, and has a process time (P_{nmf}). The starting time for each task (ST_{imf}), has an impact on the completion time (C_{imf}). The goal is to find an optimal sequence of jobs which minimizes the makespan for all factories (C_{max}) which is higher or equal to the completion time of each factory (CT_f).

In conclusion, a base model for the DPFSP is formulated as follows:

$$Z = \min_{DPFSP}(C_{max}) \tag{2.16}$$

s.t.

$$\sum_{i=1}^{N}\sum_{f=1}^{F} X_{nimf} = 1, \qquad \forall n \in N, m \in M \tag{2.17}$$

$$\sum_{n=1}^{N}\sum_{f=1}^{F} X_{nimf} = 1, \qquad \forall i \in N, m \in M \tag{2.18}$$

$$\sum_{n=1}^{N}\sum_{i=1}^{N}\sum_{m=1}^{M} X_{nimf} = A_f, \qquad \forall f \in F \tag{2.19}$$

$$C_{imf} = ST_{imf} + \sum_{n=1}^{N}(X_{nimf} \times P_{nmf}), \forall i \in N, m \in M, f \in F \tag{2.20}$$

$$C_{imf} \geq ST_{i,m-1,f} + \sum_{n=1}^{N} (X_{nimf} \times P_{nmf}), \forall i \in N, m > 1, f \in F \tag{2.21}$$

$$C_{imf} \geq ST_{i-1,mf} + \sum_{n=1}^{N} (X_{nimf} \times P_{nmf}), \forall i > 1, m \in M, f \in F \tag{2.22}$$

$$CT_f \geq \sum_{i=1}^{I} \sum_{m=1}^{M} C_{imf}, \qquad \forall f \in F \tag{2.23}$$

$$C_{max} \geq CT_f, \qquad \forall f \in F \tag{2.24}$$

$$A_f, ST_{imf}, C_{imf}, CT_f, C_{max} \geq 0 \tag{2.25}$$

$$X_{nimf} \in \{1,0\} \tag{2.26}$$

Similar to other scheduling models, the objective function aims to minimize the makespan. Constraints (2.17) and (2.18) confirm the feasibility of a schedule. The constraint set (2.19) shows that each factory must process a number of tasks. The constraint set (2.20) ensures that each schedule is directly defined by the start time and process time of the tasks. Constraints (2.21) and (2.22) confirms the sequence of machines and tasks for a schedule. The constraint set (2.23) computes the completion time for each factory and constraint set (2.24) selects one of this completion time as the makespan. At the end, the continuous and binary decision variables are defined respectively in (2.25) and (2.26).

2.2 Production scheduling under uncertainty

In the context of Industry 4.0 and IoT, advancements in production scheduling have paved the way for smart production systems (Rossit & Tohmé, 2018). Rossit et al. (2019a) conducted an empirical research study to elucidate how Industry 4.0 concepts influence production scheduling. Similarly, Zhang et al. (2019) conducted an intriguing survey where they collected real-time data and evaluated a set of job-shop scheduling models developed within the realm of Industry 4.0. Their findings provided guidelines for constructing a smart distributed scheduling model and highlighted the key distinctions between such models and traditional production scheduling approaches. Another noteworthy survey by Dolgui et al. (2019) explored the application of optimal control in production scheduling, supply chain management, and Industry 4.0-based systems. Their review paper discussed the significant contributions, applications, and recommendations for these systems, with a specific focus on existing optimal control models for production scheduling and operations management.

While Industry 4.0 practices have been recently defined and implemented in production systems (Parente et al., 2020; Rossit et al., 2019b), there has been limited attention given to developing optimization models for production systems incorporating Industry 4.0 concepts. Uncertain production scheduling models can study Industry 4.0 concepts in the production systems that may take into account disruption events such as new task arrivals (Rahmani & Ramezanian, 2016; Shen & Yao, 2015; Gao et al., 2015) or variable processing times (Framinan et al., 2019).

Shen and Yao (2015) addressed the Flexible Job-Shop Scheduling Problem (FJSP) with new task arrivals by utilizing a multi-objective evolutionary algorithm to optimize energy efficiency and job assignment stability. Gao et al. (2015) tackled the same problem using a two-stage Artificial Bee Colony (ABC) algorithm. The first stage generated an initial task schedule, and the second stage performed rescheduling upon the arrival of new tasks. Rahmani and Ramezanian (2016) studied FJSP with the potential for new task arrivals. They employed a multi-objective optimization model, with objectives including tardiness and scheduling stability. Their problem was addressed using the Variable Neighborhood Search (VNS) algorithm.

As an extension of the FSP, Fu et al. (2018) developed a model to minimize both makespan and tardiness. They considered worker learning curves in an Industry 4.0-based production system and incorporated stochastic parameters to represent machine process times and worker learning curves. To solve this model, they applied a multi-objective fireworks algorithm and compared the results with Non-dominated Sorting Genetic Algorithm (NSGA-II) (Deb et al., 2000), Multi-objective Evolutionary Algorithm based on Decomposition (MOEA/D) (Zhang et al., 2007), and a multi-start simulated annealing algorithm. Han et al. (2018) developed a blocking lot-streaming FSP with stochastic process times for an Industry 4.0-based system. They proposed a multi-objective migrating birds' optimization approach to solve the model. Framinan et al. (2019) introduced a PFSP with the possibility of variable process times for machines. They aimed to minimize makespan and employed two rescheduling strategies: continuous and periodic rescheduling.

Recent studies have generally considered multiple disruption events, including random task arrivals and machine breakdowns (Shahrabi et al., 2017; Al-Behadili et al., 2020; Ghaleb et al., 2020), leading to more complex scheduling strategies and rescheduling policies. Shahrabi et al. (2017) proposed a JSP with random task arrivals and machine failures, employing an event-driven rescheduling policy. They solved this model using the Variable Neighborhood Search (VNS) algorithm improved by reinforcement learning methods. Liu et al. (2017) developed a heuristic solution based on the Tabu Search (TS) algorithm for a mixed-shop scheduling model with the potential for new task arrivals and machine breakdowns, utilizing an event-driven rescheduling policy. Al-Behadili et al. (2020) proposed a multi-objective PFSP considering multiple disruption events. Their novel contribution was a solution algorithm based on a predictive-reactive approach that combined randomization processes with the iterated greedy algorithm. Ghaleb et al. (2020) designed an FJSP with random task arrivals and machine breakdowns, incorporating both continuous and event-driven rescheduling policies. They employed a hybrid Genetic Algorithm (GA) with three decision rules to minimize tardiness. Lastly, Gholizadeh et al. (2021) proposed a robust optimization approach for the FJSP, considering preventive maintenance, and solved it using a scenario-based GA with novel crossover and mutation operators.

These recent studies have incorporated various disruption events, such as random task arrivals and machine breakdowns, to develop more sophisticated scheduling strategies and rescheduling policies. Their contributions enhance the understanding and exploration of production scheduling within the context of Industry 4.0, paving the way for more efficient and adaptable scheduling solutions.

2.3 Sustainable production scheduling studies

Research on green and energy-efficient production scheduling has gained prominence in the context of sustainable manufacturing. Gahm et al. (2016) conducted a review that classified relevant works in sustainable scheduling into three dimensions: energy supply, energy demand and energetic coverage. Energy supply refers to the production and availability of energy sources, including electricity, fuel, and other forms of energy. It encompasses the generation, extraction, refining, processing, and distribution of energy resources to meet the demand. Energy demand refers to the quantity or amount of energy required by individuals, industries, or societies to support various activities and services. It represents the consumption or utilization of energy by end-users, such as households, businesses, transportation, and other sectors. Finally, energetic coverage refers to the extent of energy sources required to satisfy the energy demand of a particular area, region, or country. It indicates the availability and adequacy of energy resources to fulfill the energy needs of the population. The authors findings revealed a scarcity of models for energy-efficient JSP or FSP, prompting researchers to develop practical optimization models. Mansouri et al. (2016) proposed a green FSP for the first time, considering the interaction between makespan and energy consumption. They defined lower bounds and a heuristic for the problem and compared their results with the exact solver from CPLEX software.

Mokhtari and Hasani (2017) developed a multi-objective FJSP with objectives including minimizing total completion time, total energy cost, and maximizing total availability of the manufacturing system. They utilized a new version of the Strength Pareto Evolutionary Algorithm (SPEA2) (Zitzler et al., 2001) to solve their model. Wu and Sun (2018) proposed an energy-efficient FJSP with energy-saving criteria, using NSGA-II and a heuristic schedule to find Pareto-based solutions. Wang et al. (2018) presented a constructive heuristic for an energy-efficient identical parallel machine scheduling problem. They considered a set of machines that have the same processing capabilities and can perform tasks simultaneously, leading to potentially improved energy efficiency and reduced makespan. Wu and Che (2019) addressed the energy-efficient parallel machine scheduling problem, incorporating dynamic speed-scaling techniques. They proposed a memetic differential evolution algorithm with a meta-Lamarckian learning strategy for local search heuristics, comparing it with NSGA-II and SPEA2. Dai et al. (2019) proposed an energy-efficient FJSP with transportation constraints, employing an enhanced GA to generate Pareto solutions for the makespan and energy consumption objectives. Zhang et al. (2019) proposed a hybrid FSP with energy efficiency and developed a three-stage MOEA/D algorithm for multi-objective optimization.

Tirkolaee et al. (2020) introduced a variant of FSP that considered the possibility of outsourcing just-in-time delivery to minimize total cost and energy consumption concurrently. They contributed a fuzzy model to handle uncertainty and developed a self-adaptive artificial fish swarm algorithm to solve their model, comparing it with the Epsilon Constraint (EC) method (Haimes et al., 1971). Shukla et al. (2020) proposed a bi-objective model incorporating type-2 fuzzy sets to address an uncertain energy-efficient parallel machine scheduling problem. They proposed an enhanced multi-objective evolutionary algorithm in comparison with NSGA-II. Sin et al. (2020) proposed a green scheduling model considering electricity cost and preventive maintenance. They developed a hybrid multi-objective GA to find an interaction between total electricity cost and machine unavailability. Anghinolfi et al. (2020) focused on minimizing makespan and total energy consumption in an identical parallel machine scheduling problem, employing a hybrid method combining a constructive heuristic proposed by Wang et al. (2018) with local search heuristics using a greedy search. They compared their algorithm with NSGA-II and MOEA/D.

Hong et al. (2021) proposed an energy-efficient flexible FSP for a multi-cell manufacturing system with objectives including makespan, energy consumption, and total handling distance. They presented an enhanced version of MOEA/D and compared it with NSGA-II, SPEA2, and the original version of MOEA/D. Marichelvam and Geetha (2021) proposed an energy-efficient FSP under uncertainty, considering stochastic processing times. They developed a hybrid memetic algorithm with Variable Neighborhood Search (VNS) to solve the model. Finally, Jiang et al. (2022) introduced an energy-efficient FJSP, solving it with an improved Artificial Bee Colony (ABC) algorithm and comparing it with NSGA-II, Multi-Objective Bat Algorithm (MOBA), and the original version of ABC. They demonstrated the applicability of their research to complex aerospace industry components in China.

Generally, these studies collectively contribute to advancing the understanding and development of energy-efficient and sustainable production scheduling methods, with implications for various industries and sectors. However, a few studies are contributing to energy-efficient and sustainable production scheduling methods for the DPFSP.

2.4 Researches on the DPFSP

In the field of DPFSP, several significant contributions have been made by various researchers. Naderi and Ruiz (2010) were pioneers in this area, proposing two distinct Mixed Integer Linear Programming (MILP) models for the DPFSP and comparing their computational times. Additionally, they introduced two decision rules to compute the makespan. Gao and Chen (2011) developed a hybrid algorithm that combined Genetic Algorithm and Local Search Strategies (GALS) to address the DPFSP. Lin et al. (2013) modified an iterated greedy search heuristic to propose an efficient algorithm for the DPFSP. Naderi and Ruiz (2014) presented a heuristic-based scatter search approach and compared its performance with the methods proposed by Naderi and Ruiz (2010).

With the advancement of metaheuristic techniques, Bargaoui et al. (2017) introduced a novel metaheuristic inspired by chemical reactions for optimizing the DPFSP and finding the optimal makespan. Recent studies have explored alternative economic criteria as the objective function of the DPFSP. For instance, Fernandez-Viagas et al. (2018) considered the total flow-time criterion, which involves the summation of completion times for all factories. Pan et al. (2019) extended the heuristics proposed by Naderi and Ruiz (2010) by proposing an efficient decision rule for the DPFSP and enhancing the solution using iterative local search algorithms.

Similarly, Ruiz et al. (2019) developed another iterated greedy heuristic and compared its performance with the method proposed by Naderi and Ruiz (2014). Meng et al. (2019) introduced extensions to the original DPFSP by incorporating customer order constraints and proposing different classic evolutionary swarm-based algorithms. To address the issue of complexity in DPFSP, Hamzadayı (2020) developed an efficient Benders decomposition reformulation combined with local search heuristics.

In recent years, researchers have shifted their focus towards energy-efficient DPFSP due to the increasing importance of environmental sustainability. Wang and Wang (2018) introduced an energy-efficient DPFSP that simultaneously considers both makespan and energy consumption criteria. Fu et al. (2019) proposed a multi-objective brainstorm optimization algorithm for the energy-efficient DPFSP, while Wang et al. (2020) developed a multi-objective whale optimizer to solve this problem. Lu et al. (2020) proposed a sustainable DPFSP by considering a penalty coefficient for process time as a social negative impact, in addition to the makespan and energy consumption criteria. However, their sustainable DPFSP did not meet the ISO 26000 standard for contributing to employment and reducing lost workdays (Llach et al., 2015; Fathollahi-Fard et al., 2020).

Recent contributions in this field include Han et al. (2020) who proposed an energy-efficient blocking DPFSP with setup times and solved it using an improved multi-objective evolutionary algorithm incorporating Variable Neighborhood Search (VNS) and local search heuristics. Jing et al. (2021) addressed the uncertain DPFSP with stochastic process times, introducing a robust optimization approach and a hybrid metaheuristic combining iterated greedy search and local search heuristics. Huang and Gu (2021) presented another variant of the DPFSP with sequence-dependent setup times and developed a novel Biogeography-Based Optimization (BBO) algorithm to solve it. Huang et al. (2021) proposed an improved iterative greedy algorithm (IGA) to minimize total flow time in the DPFSP. Finally, Yue et al. (2023) focused on energy-efficient scheduling in the printed circuit board manufacturing industry, introducing a bi-objective mathematical model and proposing a hybrid Pareto spider monkey optimization algorithm. They also assessed its effectiveness to solve the proposed model in comparison to other multi-objective evolutionary algorithms.

Overall, these contributions demonstrate the continuous efforts to tackle the DPFSP, considering various optimization criteria and employing diverse algorithmic approaches. The shift towards energy efficiency and environmental sustainability reflects the evolving priorities in the field of production scheduling.

2.5 Research gaps and our contributions

To identify the research gaps and highlight the contributions of this Ph.D. project, Table 2.1 and Table 2.2 are provided. We focused on the most relevant works that contribute to multiple sustainability dimensions, uncertainty, or real-time scheduling. The first criterion in Table 2.1 is the configuration of the production system, followed by sustainability criteria encompassing economic, environmental, and social factors. Economic factors include makespan, flow-time, and tardiness criteria. Lastly, the contribution of different optimization methods is highlighted.

Of the approaches included in Table 2.1, we selected those related to uncertainty and real-time scheduling for further evaluation in Table 2.2. The selection of machines operating modes is also a criterion identified in some of these references. The table further categorizes uncertainty factors such as random task arrivals, machine breakdowns, and variable processing time. Rescheduling policies, including continuous, periodic, and event-driven rescheduling, are also discussed.

Based on the analysis of the literature review presented in Table 2.1 and Table 2.2, the following findings can be concluded:

- Lu et al. (2020) attempted to consider economic, environmental, and social factors simultaneously. However, their model was deterministic and did not incorporate job opportunities and lost working days.

- Jing et al. (2021) considered uncertainty in the DPFSP but did not account for random task arrivals or machine breakdowns.
- No study addressed all disruption events, including variable processing time, new task arrivals, and machine breakdowns simultaneously.
- The possibility of operating modes selection for machines has not been explored in the literature.
- Only a few studies considered rescheduling policies (Shahrabi et al. (2017); Liu et al. (2017); Framinan et al. (2019); Ghaleb et al. (2020); Al-Behadili et al. (2020)). However, these studies did not contribute to environmental and social sustainability.
- Most studies focused on advanced metaheuristics, while the development of reformulation techniques was rarely explored.

Table 2.1 Summary of relevant studies based on sustainability criteria

Reference	Configuration of production systems	Sustainability criteria			Proposed solution algorithm
		Economic	Environmental	Social	
Shen and Yao (2015)	FJSP	✓	✓	-	Modified evolutionary algorithm
Gao et al., (2015)	FJSP	✓	-	-	Two-stage ABC
Rahmani and Ramezanian (2016)	FJSP	✓	-	-	VNS
Shahrabi et al., (2017)	JSP	✓	-	-	VNS with reinforcement learning
Liu et al., (2017)	Mixed-shop	✓	-	-	TS
Wang and Wang (2018)	DPFSP	✓	✓	-	Knowledge-based cooperative algorithm
Fu et al., (2018)	FSP	✓	-	-	Multi-objective fireworks algorithm
Han et al., (2018)	FSP	✓	-	-	Multi-objective migrating birds' optimization
Fu et al., (2019)	DPFSP	✓	✓	-	Brain storm optimization
Framinan et al., (2019)	PFSP	✓	-	-	-
Wang et al., (2018)	DPFSP	✓	✓	-	Multi-objective whale optimization algorithm
Han et al., (2020)	BFS	✓	✓	-	Improved multi-objective evolutionary algorithm
Lu et al., (2020)	DPFSP	✓	✓	✓	Multi-objective memetic algorithm
Al-Behadili et al., (2020)	PFSP	✓	-	-	Iterated greedy algorithm
Ghaleb et al., (2020)	FJSP	✓	-	-	Hybrid GA and Heuristics
Gholizadeh et al., (2021)	FJSP	✓	-	-	Scenario-based GA
Mansouri et al., (2016)	FSP	✓	✓	-	Lower bounds and a heuristic
Mokhtari and Hasani (2017)	FJSP	✓	✓	-	Improved SPEA2
Wu and Sun (2018)	FJSP	✓	✓	-	NSGA-II with heuristics
Wang et al., (2018)	Parallel machine	✓	✓	-	Constructive heuristic
He and Sun (2013)	FJSP	✓	-	-	-
Jing et al., (2021)	DPFSP	✓	-	-	Iterated greedy with local search
Wu and Che (2019)	Parallel machine	✓	✓	-	Hybrid memetic differential evolution algorithm

(Continued)

Reference	Configuration of production systems	Sustainability criteria			Proposed solution algorithm
		Economic	Environmental	Social	
Dai et al., (2019)	FJSP	✓	✓	-	Enhanced GA
Zhang et al., (2019)	Hybrid FSP	✓	✓	-	Three-stage MOEA/D
Tirkolaee et al., (2020)	FSP	✓	✓	-	Self-adaptive artificial fish swarm algorithm
Shukla et al., (2020)	Parallel machine	✓	✓	-	Enhanced multi-objective evolutionary algorithm
Sin et al., (2020)	Parallel machine	✓	✓	-	Hybrid multi-objective GA
Anghinolfi et al., (2020)	Parallel machine	✓	✓	-	Greedy search with local search
Hong et al., (2021)	FSP	✓	✓	-	Improved MOEA/D
Marichelvam and Geetha (2021)	FSP	✓	✓	-	Hybrid memetic with VNS
Jiang et al., (2022)	FJSP	✓	✓	-	Improved ABC
Yue et al. (2023)	DPFSP	✓	✓	-	Improved Pareto-spider monkey optimization
This research	DPFSP	✓	✓	✓	Reformulations, Heuristics and Metaheuristics

To address these research gaps, this Ph.D. project aims to develop a comprehensive framework for the smart and sustainable DPFSP. The project will contribute to the development of different rescheduling policies, scheduling strategies, and robust optimization techniques. For the first time, it will consider different operating modes for each machine, ranging from manual to automatic modes. Additionally, the project will incorporate social sustainability criteria into the DPFSP, including job opportunities created and workdays lost. Various uncertainty factors, such as variable processing time, new task arrivals, and machine breakdowns, will be simultaneously considered. The project will develop a deterministic model and an uncertain model using continuous and event-driven rescheduling policies. Finally, a robust optimization model will be introduced to analyze all possible disruptions across a full range of probabilistic scenarios. Innovative solution methods, including reformulations, heuristics, and metaheuristics such as ALNS, will be developed within this Ph.D. project.

Table 2.2 Relevant studies based on uncertainty and real-time scheduling criteria

Reference	Uncertainty			Operating mode selection	Rescheduling policies		
	Random task arrival	Machine's breakdown	Variable process time		Continuous rescheduling policy	Periodic rescheduling policy	Event-driven rescheduling policy
Shen and Yao (2015)	✓	-	-	-	-	-	-
Gao et al., (2015)	✓	-	-	-	-	-	-
Rahmani and Ramezanian (2016)	✓	-	-	-	-	-	-
Shahrabi et al., (2017)	✓	✓	-	-	-	-	✓
Liu et al., (2017)	✓	✓	-	-	-	-	✓
Fu et al., (2018)	-	-	✓	-	-	-	-
Han et al., (2018)	-	-	✓	-	-	-	-
Framinan et al., (2019)	-	-	✓	-	✓	✓	
Al-Behadili et al., (2020)	✓	✓	-	-	-	-	✓
Ghaleb et al., (2020)	✓	✓	-	-	✓	-	✓
Gholizadeh et al., (2021)	-	✓	✓	-	-	-	-
He and Sun (2013)	-	✓	-	-	✓	-	-
Jing et al., (2021)	-	-	✓	-	-	-	-
Wu and Che (2019)	-	-	✓	-	-	-	-
Tirkolaee et al., (2020)	✓	-	✓	-	-	-	-
Shukla et al., (2020)	-	-	✓	-	-	-	-
Sin et al., (2020)	-	✓	-	-	-	-	-
Marichelvam and Geetha (2021)	-	-	✓	-	-	-	-
This research	✓	✓	✓	✓	✓	-	✓

CHAPTER 3

SUSTAINABLE DISTRIBUTED PERMUTATION FLOW SHOP SCHEDULING MODEL BASED ON A TRIPLE BOTTOM LINE CONCEPT

Amir M. Fathollahi-Fard, Lyne Woodward and Ouassima Akhrif,

Department of Electrical Engineering, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper published in the *Journal of Industrial Information Integration,* September 2021

Abstract

Based on a triple bottom line concept, sustainable development is characterized by the simultaneous pursuit of economic, environmental and social goals. The implementation of this concept in production scheduling can result in the resolution of a sustainable Distributed Permutation Flow Shop Scheduling Problem (DPFSP). The present study conceptually shifts an energy-efficient DPFSP to a sustainable DPFSP, simultaneously contributing to economic, environmental and social improvements. The study aims not only to minimize the total energy consumption related to production, but also, to maximize, for the first time, the social factors linked to job opportunities and lost working days. Different production centers and operating modes such as manual and automatic modes are considered as new suppositions to establish a sustainable DPFSP. In this regard, a novel multi-objective mixed integer linear model is developed. To manage the high complexity of the proposed model, a novel multi-objective learning-based heuristic is established, as an extension of the Social Engineering Optimizer (SEO). The applicability of the proposed model is determined in the context of the wood industry in Canada. Several simulated tests are considered to verify the model. The proposed heuristic is compared with one of the other well-known, recent and state-of-the art methods. In order to guarantee a fair comparison, the Taguchi method is used to tune the parameters of the algorithms. Finally, sensitivity analyses are done to assess the efficiency of the proposed model.

Keywords: Triple bottom line approach, production scheduling, distributed permutation flow shop scheduling problem, learning-based heuristic, social engineering optimizer.

3.1 Introduction

With environmental sustainability and social responsibility trending as a means of tackling environmental deterioration, economic performance, social equity and other sustainability issues, the Triple Bottom Line (TBL) approach has become an active research topic in the supply chain, logistics and production management fields (Fathollahi-Fard et al., 2020). A sustainable production system is academically defined as a production system that takes economic, environmental and social factors into account (Lu et al., 2020). Merging the concept of TBL with production systems opens up several new avenues for researches in terms of developing optimization models and algorithms for production planning (Shao et al., 2020; Han et al., 2020; Mansouri et al., 2016). In this context, the present work aims to find a way to model a sustainable production system for a wood processing company in Canada. According to the government of Canada[1], production by the wood industry contributed a total of \$19.8 billion to the country's GDP in 2013. Hence, Canada has the largest market share of the wood industry in the world. However, based on the sustainable development paradigm and the TBL criteria, the Canadian wood industry's production systems must be redesigned to holistically include economic, environmental and social factors.

Many companies operating in the Canadian wood industry usually focus on economic performance, while ignoring social and environmental issues. Environmental sustainability is crucial as reports[1] indicate that in Canada, this industry is responsible for half of all carbon emissions. Machines in production centers use non-renewable energy, which represents a challenge in terms of meeting cleaner production goals.

---

[1] https://www.nrcan.gc.ca/home

The quality of human life and social sustainability are closely intertwined (Fathollahi-Fard et al., 2020; Xu et al., 2020). In the literature, job opportunities and reduction of lost working days are two pivotal factors essential for achieving social sustainability in line with the guidelines of ISO 26000 (Llach et al., 2015). In manufacturing industries, each machine operating in a specific mode, whether manual or automatic, requires skilled operators for its operation. However, when there's a shift in the underlying operating mode of these machines, the production system can face disruptions, leading to lost workdays. These lost days are often dedicated to training and updating of operators' skills and knowledge in new advanced operating modes. To address these challenges, this study presents a multi-objective optimization model aimed at solving the sustainable Distributed Permutation Flow Shop Scheduling Problem (DPFSP).

The literature on the DPFSP is very rich, and provides many optimization models and algorithms (Shao et al., 2020; Han et al., 2020). Obviously, the solution complexity of the DPFSP is very high, and is classified as NP-hard. Therefore, many heuristics have been developed to deal with this (Lu et al., 2016; Wang et al., 2020; Ye et al., 2020; Zhu et al., 2020; Duarte et al., 2020). The multi-objective optimization model proposed in the present work is even more complex than what is seen in most current studies (Anghinolfi et al., 2020; Abreu et al., 2020; Ghaleb et al., 2020), as it integrates social and environmental factors to the DPFSP, as well as economic factors such as the makespan and the total cost. Real-life constraints of the industry such as multiple production centers and operating modes selection are also included. To the best of our knowledge, no existing optimization algorithm is suitable for solving our complex model due to the theory of no free lunch (Wolpert, & Macready, 1997). Based on this challenge, this study innovates a new single solution heuristic as a variation of the Social Engineering Optimizer (SEO) (Fathollahi-Fard et al., 2018), using local search heuristics and a learning-based operator.

All in all, the present study makes the following contributions to the literature:

- A sustainable DPFSP based on the TBL concept is formulated as a new multi-objective mixed integer linear programming model;

- A novel learning-based SEO is heuristically introduced for solving the proposed problem;
- An industrial example of the wood industry in Canada is proposed to show the applicability of the simulation results.

The rest of this chapter is broken down as follows: Section 3.2 assesses recent and relevant studies in the DPFSP domain, identifying any research gaps. Section 3.3 establishes the definition of sustainable DPFSP and outlines the problem's formulation. Section 3.4 details the solution representation of our optimization model, introduces our new heuristic procedures and presents the proposed learning-based SEO approach. Section 3.5 offers an industrial example of our model and conducts simulation tests, facilitating a comprehensive analysis of both the model and the solutions generated. Section 3.6 discusses insights drawn from our results and contributions, providing an in-depth analysis that leads to practical insights and managerial solutions. Section 3.7 presents conclusions and offers recommendations for production system managers. It also discusses our research findings and outlines potential future research directions.

3.2 Literature review

The DPFSP is a type of distributed production system in which production tasks are first assigned to different production centers, following which the system scheduling is planned and executed (Lu et al., 2016; Wang et al., 2020). The DPFSP is academically an extension of the Permutation Flow Shop Scheduling Problem (PFSP) (Ye et al., 2020; Naderi & Ruiz, 2010). However, the general flow shop scheduling model schedules tasks only for one production center (Zhu et al., 2020). Although many studies have applied the DPFSP to many industrial applications, such as automobile production and petrochemicals, most related studies have only looked at economic factors such as the makespan and tardiness, while ignoring sustainability criteria including energy-consumption and social benefits.

This section comprises a review of the most relevant works that have dealt with the DPFSP during the last decade. In 2010, Naderi and Ruiz (2010) were the first to study the DPFSP.

They solved a Mixed Integer Linear Programming (MILP) model with a view to reducing the makespan, using two heuristics for the assignment of designed production centers. Then, in 2011, Gao and Chen (2011) proposed a Genetic Algorithm with Local Search Strategies (GALS) to address the DPFSP, taking the makespan into account. In 2013, Lin et al. (2013) studied the DPFSP using a modified iterated greedy search algorithm. Similarly, in 2014, Naderi and Ruiz (2014) developed a novel scatter search heuristic for this problem and compared its efficiency to that of other existing methods. In 2017, Bargaoui et al. (2017) applied an optimization algorithm inspired by chemical reactions to address the DPFSP with the makespan criterion.

Two economic factors used in the literature are the makespan criterion which computes the maximum time of completion between all production centers, and the total flow-time criterion, which is the summation of the completion time for all these centers. In 2018, consideration of the total flow time criterion as an objective function was first proposed by Fernandez-Viagas et al. (2018). Then, in 2019, Pan et al. (2019) solved the DPFSP with heuristic-based local search algorithms. In the same year, Ruiz et al. (2019) proposed simplified iterated greedy heuristics to solve the DPFSP, while Meng et al. (2019) developed the DPFSP to reduce the makespan under customer order constraints by the use of evolutionary and swarm-based optimization algorithms.

Consideration of environmental sustainability along with economic factors has recently appeared in the literature. In (Wang, & Wang, 2018), Wang and Wang proposed, for the first time, an energy-efficient DPFSP to optimize the makespan and energy consumption simultaneously. Fu et al. (2019) solved a stochastic energy-efficient DPFSP by a brain storm optimization heuristic. Wang et al. (2020) developed a multi-objective whale optimization algorithm to solve the energy-efficient DPFSP. Last, but not least, Lu et al. (2020) proposed the concept of a sustainable DPFSP, taking into consideration the energy consumption and a penalty coefficient for the process time. They defined this penalty function as a negative social factor. However, it does not meet the social sustainability criterion under the TBL and ISO 26000 (Llach et al., 2015; Fathollahi-Fard et al., 2020). They solved the problem using a multi-objective memetic optimization algorithm. It goes without saying that there are several other variants of the DPFSP (Xu et al., 2014; Jing et al., 2021), including for instance, the blocking DPFSP (Han et al., 2020), preventive maintenance (Ye et al., 2020), and the no-wait DPFSP (Zhu et al., 2020).

Table 3.1 Summary of the literature review for DPFSP studies

Paper	Year	Sustainability factors			Solution algorithm
		Economic	Environmental	Social	
Naderi & Ruiz (2010)	2010	✓	-	-	Heuristics
Gao & Chen (2011)	2011	✓	-	-	GALS
Lin et al. (2013)	2013	✓	-	-	Modified iterated greedy search
Naderi & Ruiz (2014)	2014	✓	-	-	Scatter search
Xu et al. (2014)	2014	✓	-	-	Hybrid immune algorithm
Bargaoui et al. (2017)	2017	✓	-	-	Chemical reaction algorithm
Fernandez-Viagas et al. (2018)	2018	✓	-	-	Evolutionary search
Wang & Wang (2018)	2018	✓	✓	-	Knowledge-based cooperative algorithm
Pan et al. (2019)	2019	✓	-	-	Local search heuristic
Ruiz et al. (2019)	2019	✓	-	-	Simplified iterated greedy search
Meng et al. (2019)	2019	✓	-	-	Swarm-based evolutionary algorithm
Fu et al. (2019)	2019	✓	✓	-	Brain storm optimization
Wang et al. (2020)	2020	✓	✓	-	Whale swarm algorithm
Lu et al. (2020)	2020	✓	✓	✓	Memetic optimization algorithm
Jing et al. (2021)	2021	✓	-	-	Local search-based metaheuristics
Huang & Gu (2021)	2021	✓	-	-	Biogeography-based optimization algorithm
This study	2021	✓	✓	✓	Learning-based SEO with local search

Table 3.1 presents a summary of the literature review and collects all the DPFSPs related to sustainability factors, including economic, environmental and social factors, as well as the solution algorithm. From this table, the following research gaps can be identified:

- Only one study by Lu et al., (2020) has considered the triple bottom line concept in modeling a sustainable DPFSP. However, the job opportunity and lost working days were not considered.
- No study has applied an SEO or any version of this algorithm to the area of DPFSP.

In terms of research gaps, only one study (Lu et al., 2020) considered social factors using a penalty coefficient associated with the task process times. However, as can be seen in the ISO 2600 guidelines respecting social responsibility in the production and supply chain systems to improve humans' life quality (Benoît et al., 2010; Llach et al., 2015), job opportunities and lost working days are two of the main factors that must be considered in order to achieve social sustainability. In this regard, a novel multi-objective MILP is developed to minimize the makespan and energy consumption while maximizing social benefits. To get our DPFSP closer to real production systems such as those in the Canadian wood industry, the proposed problem assumes that centers are non-identical as they handle different forest products. This problem also considers different operating modes for machines which have a high impact on environmental and social factors. New operating modes can increase the speed of operation of tasks, but at the cost of increasing the energy consumed and creating fewer job opportunities for workers in comparison to traditional production systems. To the best of the authors' knowledge, no similar study has considered these items simultaneously in order to establish a sustainable DPFSP. Another novelty of this paper is the development of a new optimizer for solving our mathematical model. This study proposes a new SEO version (Fathollahi-Fard et al., 2018) created with the help of learning-based operators and local search-based heuristics to solve our multi-objective optimization problem. The proposed algorithm is able to generate higher-efficient Pareto-based solutions than what are obtained with the general version of SEO and other state-of-the art methods in the literature.

3.3 Proposed problem

This section starts by defining the notations used for the mathematical modeling of the proposed sustainable DPFSP as follows:

Indices:

f	Index of production centers, $f \in \{1, 2, \dots, F\}$
m	Index of machines in each center, $m \in \{1, 2, \dots, M\}$
n	Index of tasks, $n \in \{1,2, \dots, N\}$
t	Index of operating modes, $t \in \{1,2, \dots, T\}$
i	Index of task positions in a schedule, $i \in \{1,2, \dots, N\}$

Parameters:

B	Maximum budget for the machines with their operating modes and the salary of workers for all the production centers
CO_{mtf}	Cost of machine *m* using operating mode *t* in the production center *f*
JO_{mtf}	Job opportunities created by the use of machine *m* with operating mode *t* in the production center *f*
CJ_{mtf}	Salary of operators working on machine *m* with operating mode *t* per unit of time in the production center *f*
LD_{mtf}	Lost days due to the use of machine *m* with operating mode *t* in the production center *f*
MW	Maximum allowable ratio of broken products in all the production centers
RW_{mtf}	Ratio of broken products when machine *m* using operating mode *t* are used in the production center *f*
O_{nmtf}	Operation of task *n* on machine *m* with operating mode *t* in the production center *f*
P_{nmtf}	Process time of operation O_{nmtf}
IEC_{mtf}	Idle energy consumption of machine *m* with operating mode *t* per unit of time in the production center *f*
UEC_{mtf}	Useful energy consumption of machine *m* with operating mode *t* per unit of time in the production center *f*

EC_{mtf} Energy consumption due to implementing machine *m* with operating mode *t* per unit of time in the production center *f*

WJ Weight of job opportunities

WL Weight of lost working days

Decision variables:

A_f Number of tasks assigned to the production center *f*

Y_{mtf} If the machine *m* is using the operating mode *t* in the production center *f*, 1; otherwise, 0

ST_{imtf} Starting time of the task at position *i* on machine *m* using operating mode *t* in the production center *f*

X_{nimtf} If the task *n* is set at position *i* on machine *m* with the use of operating mode *t* in the production center *f*, 1; otherwise, 0

T_{nmtf} Idle time of the operation of task *n* on machine *m* with operating mode *t* in the production center *f* (O_{nmtf})

C_{imtf} Completion time of a task at position *i* on machine *m* with operating mode *t* in the production center *f*

CT_f Time for completing tasks in the production center *f*

C_{max} Maximal completion time for all the production centers

From the description of the proposed problem, there are *N* tasks distributed across *F* non-identical production centers. Each center has *M* different machines with *T* operating modes and follows a PFSP conceptually. For each task, there are *O* operations. These operations are handled one by one for the assigned production center. All the production centers are able to perform all the tasks. When the scheduling starts, all the machines and centers are available. After a task is assigned to a production center, the task must be processed at that center, and cannot be transferred to another one. No interruption is allowed in the proposed production system. The process time (P_{nmtf}) for the operation (O_{nmtf}) of the tasks is different based on the production centers, machines and operating modes. For each machine, there are some operating modes from manual to automatic ones which change the speed of the operations, the energy consumption and the social factors linked to job opportunities for operators, and work days lost while learning this technology and updating the workers' knowledge. Next, the criteria of TBL used in the definition of our problem, i.e., economic sustainability, environmental sustainability and social sustainability, are illustrated followed by our proposed mathematical model.

3.3.1 Economic sustainability

In most production scheduling models, the makespan (C_{max}) is the only economic criterion (Naderi, & Ruiz, 2014, Peng et al., 2019; Ruiz et al., 2019). This criterion reflects the benefit of a production system or its economic value. The present study is not limited to the achievement of economic sustainability. It considers not only the makespan, but also the worker salaries and the production rates of operating modes used. Let us assume that a company supports the total cost of a production system and has a maximum budget (B). This company must consider the costs of purchasing machines with updated operating modes (CO_{mtf}). Last but not least, workers' salaries also vary with the production centers, the machines and the operating modes (CJ_{mtf}).

3.3.2 Environmental sustainability

Environmental pollution in production operations is certainly the main culprit in the context of global warming and climate change in developed countries like Canada. To control environmental pollution in production systems and supply chains (Sabuj et al., 2021; Karmaker et al., 2020), the International Organization for Standardization (ISO) proposed the ISO 14000 standard for environmental sustainability management (Corbett et al., 2001). Having cleaner production is particularly a concern in the lumber industry, considered to be the leader of environmental pollution in Canada. Compared to traditional technologies which are generally based on the use of non-renewable energies, new production technologies consume fewer non-renewable energy resources. In this regard, recent studies proposed the energy-efficient DPFSP as a solution for this challenge (Wang, & Wang, 2018; Fu et al., 2019; Wang et al., 2020). To achieve energy consumption sustainability, this study not only considers the energy consumption of working time (UEC_{mtf}) and idle time (IEC_{mtf}), but also the energy consumption related to the implementation of the technology of turning the machines on and off (EC_{mtf}). The waste reduction is another criterion for achieving the environmental sustainability. In the proposed problem, only one operation mode must be implemented on each machine. Wastes are different for each operating mode such as manual or automatic one (RW_{mtf}), and with regards to the concept of economic sustainability, the maximum allowable wastes should not be exceeded by the machines under these operating modes (MW).

3.3.3 Social sustainability

Social sustainability involves many factors linked to the work environment, healthcare and social development. In the ISO 26000 standard used by governments and business networks to achieve social responsibility (Benoît et al., 2010), there is a guideline, the SA8000, that considers the job opportunities and lost days for injuries of workers (Llach et al., 2015). It should be noted that social sustainability is not limited to only these two factors as consumer risk and local business development are two other common criteria used in relevant works (Marimuthu et al., 2021).

For the first time in the area of DPFSP, the number of operators working on a machine using a particular operating mode on a machine in a production center is considered explicitly (JO_{mtf}). The number of work days lost due to the implementation of a new operation mode on a machine (LD_{mtf}) is considered as another social factor. The lost working days represent in fact the time needed to teach operators working on this new operating mode. These social factors are weighted (WJ and WL) in the third objective function which aims to achieve social sustainability.

3.3.4 Mathematical model

Generally, the proposed sustainable DPFSP aims to find, for each production center, the optimal number of tasks allocated (A_f), the time needed to complete the tasks (CT_f), the optimal allocation of operating modes to machines (Y_{mtf}), the optimal sequence of tasks (X_{nimtf}) and other optimal values of the decision variables defined earlier. The proposed mathematical model is described as follows:

$$Z_1 = min(C_{max}) \tag{3.1}$$

$$\begin{aligned} Z_2 = min\Bigg(&\sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}(Y_{mtf} \times EC_{mtf}) \\ &+ \sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}\sum_{n=1}^{N}\sum_{i=1}^{N}(X_{nimtf} \times UEC_{mtf} \times P_{nmtf}) \\ &+ \sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}\sum_{n=1}^{N}(T_{nmtf} \times IEC_{mtf})\Bigg) \end{aligned} \tag{3.2}$$

$$\begin{aligned} Z_3 = max\Bigg(&WJ \times \sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}(Y_{mtf} \times JO_{mtf}) \\ &- WL \times \sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}(Y_{mtf} \times LD_{mtf})\Bigg) \end{aligned} \tag{3.3}$$

s.t.

$$\sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}(Y_{mtf} \times JO_{mtf} \times CJ_{mtf}) + \sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}(Y_{mtf} \times CO_{mtf}) \leq B \tag{3.4}$$

$$\sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}(Y_{mtf} \times RW_{mtf}) \leq MW \tag{3.5}$$

$$\sum_{i=1}^{N}\sum_{f=1}^{F}X_{nimtf} = 1, \qquad \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T} \tag{3.6}$$

$$\sum_{n=1}^{N}\sum_{f=1}^{F}X_{nimtf} = 1, \qquad \forall i \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T} \tag{3.7}$$

$$\sum_{n=1}^{N}\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{t=1}^{T}(X_{nimtf}) = A_f, \qquad \forall f \in \mathcal{F} \tag{3.8}$$

$$\sum_{n=1}^{N}\sum_{i=1}^{N}X_{nimtf} \leq N \times Y_{mtf}, \qquad \forall m \in \mathcal{M}, t \in \mathcal{T}, f \in \mathcal{F} \tag{3.9}$$

$$\sum_{t=1}^{T}Y_{mtf} = 1, \qquad \forall m \in \mathcal{M}, f \in \mathcal{F} \tag{3.10}$$

$$\sum_{i=1}^{N}ST_{imtf} \leq \left(\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{f=1}^{F}P_{nmtf}\right) \times Y_{mtf}, \forall m \in \mathcal{M}, t \in \mathcal{T}, f \in \mathcal{F} \tag{3.11}$$

$$C_{imtf} = ST_{imtf} + \sum_{n=1}^{N}(X_{nimtf} \times P_{nmtf}), \forall i \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, f \in \mathcal{F} \tag{3.12}$$

$$C_{imtf} \geq ST_{i,m-1,tf} + \sum_{n=1}^{N}(X_{nimtf} \times P_{nmtf}), \forall i \in \mathcal{N}, m \in \mathcal{M}, m > 1, t \in \mathcal{T}, f \in \mathcal{F} \tag{3.13}$$

$$C_{imtf} \geq ST_{i-1,mtf} + \sum_{n=1}^{N}(X_{nimtf} \times P_{nmtf}), \forall i > 1, i \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, f \in \mathcal{F} \tag{3.14}$$

$$T_{nmtf} = \sum_{i>1}^{N}(C_{imtf} - C_{i-1,mtf} - (X_{nimtf} \times P_{nmtf})), \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, f \in \mathcal{F} \tag{3.15}$$

$$CT_f \geq \sum_{i=1}^{I}\sum_{m=1}^{M}\sum_{t=1}^{T}C_{imtf}, \qquad \forall f \in \mathcal{F} \tag{3.16}$$

$$C_{max} \geq CT_f, \qquad \forall f \in \mathcal{F} \tag{3.17}$$

$$A_f, ST_{imtf}, C_{imtf}, CT_f, C_{max}, T_{nmtf} \geq 0, \qquad \forall n, i \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, f \in \mathcal{F} \tag{3.18}$$

$$Y_{mtf}, X_{nimtf} \in \{1,0\}, \quad \forall n, i \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, f \in \mathcal{F} \tag{3.19}$$

Equations (3.1) to (3.3) represent the objective functions which are limited by constraints (3.4) to (3.19). The optimal solution is found by minimizing the makespan (Equation (3.1)) and the energy consumption (Equation (3.2)) while maximizing the social benefits (Equation (3.3)). In the second objective function, the energy consumption includes the energy required to implement an operating mode in a machine, the energy used to process a task on a machine and the energy consumed by a machine during an idle period of time when a task is pending. In the last objective function, we consider two distinct social benefit criteria: job opportunities and lost working days. These criteria have different units and do not share the same range. Our objective is to maximize job opportunities while minimizing lost working days. To achieve this, we employ a weighted sum approach to combine these social criteria into a single objective. The weights assigned to these criteria are adjusted to bring them into a comparable range, as they naturally differ in scale.

The constraint set (3.4) concerns the maximum budget available to cover the salary of the operators and the cost associated with the implementation of operating modes on the machines. The maximum ratio of broken products or waste authorized in all production centers is considered in the set of constraints (3.5). Constraints (3.6) and (3.7) show that each task must have a unique schedule. The constraint set (3.8) guarantees that the required number of tasks is assigned in each production center. The constraint set (3.9) shows the relationship between the allocation of tasks to production centers and the operating mode selection for machines. The constraint set (3.10) ensures that each machine is assigned to one operating mode such as manual or automatic. The constraint set (3.11) relates the start time of tasks to the total operating time of all machines in all production centers. The constraint set (3.12) shows that the full time of a task is defined by its start and processing times. Constraints (3.13) and (3.14) show the relationship between machine schedules and tasks in a sequence. Constraint (3.15) computes the idle time of the machines. The constraint set (3.16) limits the maximum time allowed to complete all tasks in a production center while the constraint set (3.17) ensures that the makespan of all production centers is less than or equal to the maximum allowable execution time. Finally, the constraints (3.18) and (3.19) define the feasible set of values of the decision variables in the model.

3.3.5 Numerical example

In order to show that the proposed optimization model has a feasible solution and to numerically illustrate the proposed sustainable DPFSP, an example with 4 tasks (J_1, J_2, J_3 and J_4), 2 production centers (F_1 and F_2), 2 machines (M_1 and M_2) and 2 operating modes (T_1 and T_2), is provided. For these operating modes, we can assume that they are only two modes of manual or automatic for each machine in each production center. Table 3.2 provides the data used for task processing time, cost of machines, job opportunities, lost days, operators' salaries, idle and utile energy consumption rate, and energy consumption for switching machines on and off. The maximum budget of the company is set to 0.5 million dollar and the maximum allowable number of broken products is set to 30 percent in this example. Finally, the weights for social factors, including job opportunities and lost working days, are set at 0.9 and 0.1 respectively.

Table 3.2 Processing time of tasks and other parameters values

Tasks	Unit	Production center F_1				Production center F_2			
		Machine M_1		Machine M_2		Machine M_1		Machine M_2	
		T_1	T_2	T_1	T_2	T_1	T_2	T_1	T_2
J_1	Hour	4	5	6	5	3	4	6	6
J_2	Hour	3	6	4	4	4	6	3	4
J_3	Hour	5	4	2	2	6	5	2	3
J_4	Hour	2	3	4	6	3	4	5	4
Implementation cost (CO_{mtf})	$	12×10^4	15×10^4	13×10^4	9×10^4	12×10^4	14×10^4	11×10^4	12×10^4
Job opportunities (JO_{mtf})	Person	2	3	4	2	4	6	3	5
Salary of operators (CJ_{mtf})	$	10	8	12	10	10	9	12	8
Lost days (LD_{mtf})	Days	14	10	21	14	10	14	12	8
Ratio of broken products (RW_{mtf})	Scalar	0.08	0.02	0.05	0.09	0.05	0.07	0.06	0.07
Idle energy consumption rate (IEC_{mtf})	BTU per hour (*[2])	8.98×10^5	9.8×10^5	10.2×10^5	8.75×10^5	8.75×10^5	8.55×10^5	8.25×10^5	8.6×10^5
Utile energy consumption rate (UEC_{mtf})	BTU per hour	5.36×10^5	5.4×10^5	6.4×10^5	5.2×10^5	4.6×10^5	3.6×10^5	5.1×10^5	5.5×10^5
Energy consumption for implementation (EC_{mtf})	BTU	30.2×10^5	28.4×10^5	26.2×10^5	25.4×10^5	27.2×10^5	26.8×10^5	24.8×10^5	26.2×10^5

This numerical example demonstrates the feasibility of the proposed optimization model, as it possesses an optimal solution. In solving this example, our focus is primarily on optimizing the first objective, i.e., the makespan. The values of the other objectives are subsequently computed after this objective. An investigation on this optimal solution shows that it allocates the first operating mode (T_1) to the first machine (M_1) in the first production center (F_1). In the same center, the second operating mode (T_2) is assigned to the second machine (M_2). In the second production center (F_2), the first operating mode is selected to be used for both machines. Therefore, the decision variables are $Y_{111} = Y_{221} = Y_{112} = Y_{212} = 1$ while the others are zero.

[2] British Thermal Unit (BTU), note that 10^{15} BTU equals to 1055×10^{18} joules

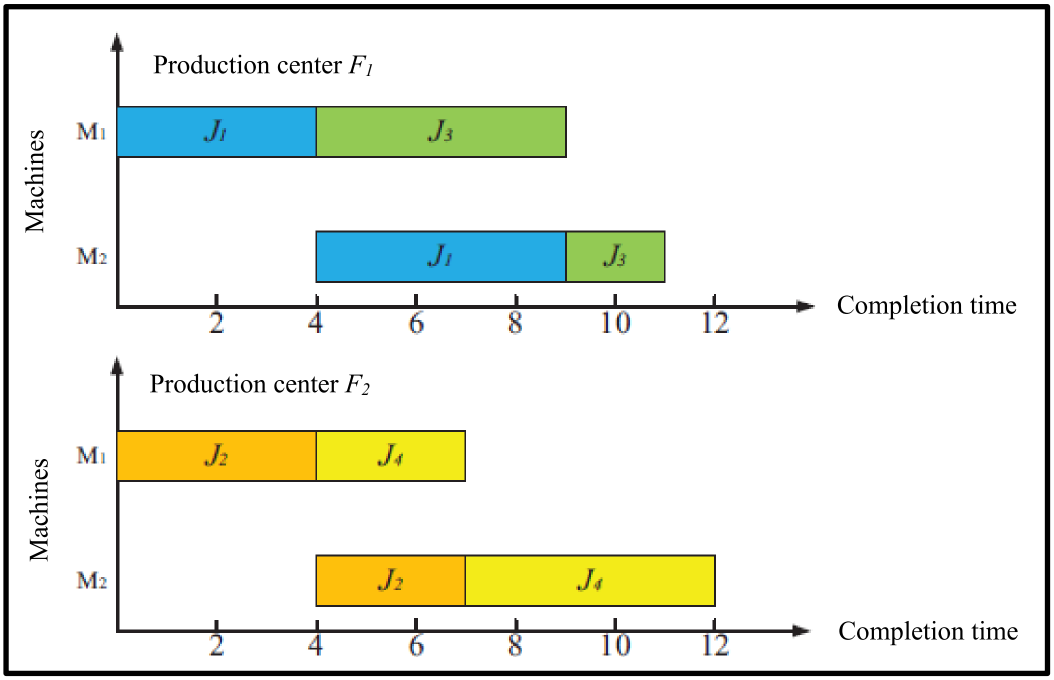


Figure 3.1 Presentation of the optimal solution

The permutation of the tasks [J_1, J_2, J_3, J_4] of the optimal solution is shown in Figure 3.1. The tasks J_1 and J_3 are assigned to the first production center, while the tasks J_2 and J_4 are assigned to the second production center. The completion times are 11 and 12 hours respectively in the first and second centers. Based on these outputs, the makespan is 12 hours, the total energy consumption is 33 324 000 BTUs and finally the social criteria value is 4.9.

3.4 Proposed algorithm

As mentioned earlier, the classical version of the DPFSP is NP-hard. The proposed DPFSP which includes three conflicting objectives and real-life constraints such as, for example, the maximal budget, is more complex than most of the existing studies. To tackle this optimization model, this study develops a new metaheuristic that is an extension of the recently proposed SEO approach (Fathollahi-Fard et al., 2018). This extension includes new learning-based operators and local search-based techniques added to the SEO for solving our multi-objective problem.

The SEO algorithm was chosen as the base method because of its high computational time efficiency in solving NP-hard problems such as routing optimization (Mojtahedi et al., 2021) and truck scheduling problems (Fathollahi-Fard et al., 2019). Furthermore, although many recently developed optimizers such as the immune (Xu et al., 2014), chemical reaction (Bargaoui et al. 2017), whale swarm (Wang et al., 2020), and brain storm (Fu et al., 2019) algorithms have been studied in the field of DPFSP, to the best of our knowledge, no study has applied SEO so far in this area. The flowchart shown in Figure 3.2 presents the general framework of the original SEO algorithm.

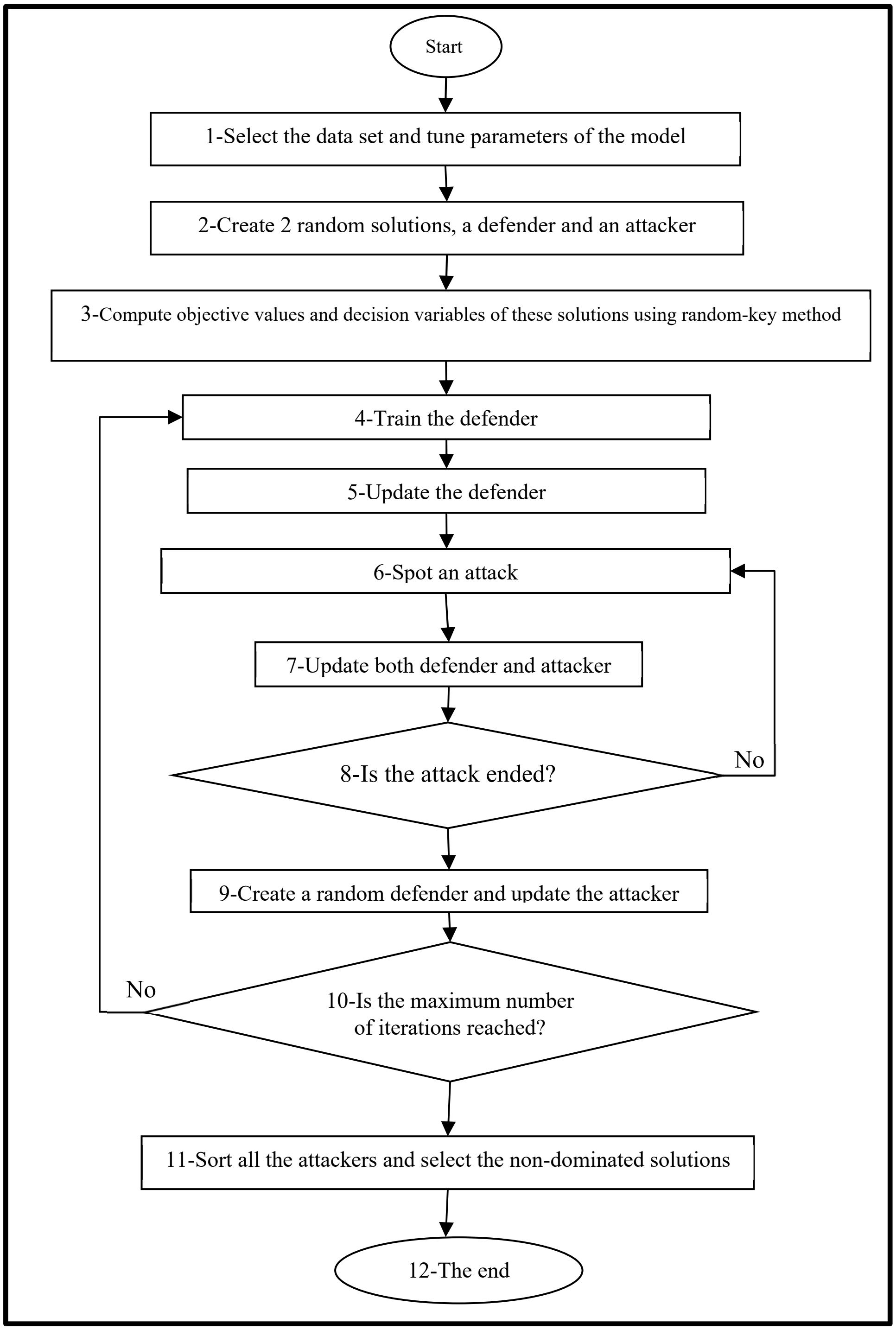


Figure 3.2 Flowchart of the original SEO algorithm

Based on social engineering rules, the approach starts with two initial solutions determined randomly (step 2). Based on the concept of Pareto dominance, the algorithm sorts the solutions and defines the attacker and the defender (step 3). To train the defender (step 4), the attacker copies a percentage of the defender. If this newly trained defender is dominant, the last defender is updated with this new one (step 5). Then, the social engineering attacks start (step 6): new attackers and defenders are generated and compared to the previous ones. If a new defender is able to dominate the attacker, their positions are swapped (step7). Finally, the best solution is saved as the attacker and sent to the list of Pareto solutions while a new random solution replaces the defender (step 9). Once the maximum number of iterations has been reached (step 10), the list of Pareto solutions is sorted and the non-dominated solutions are found (step 11).

The main difference of the proposed algorithm with the original multi-objective SEO is in steps 4 and 6 where a learning-based operator and a local search technique are developed to improve the performance of SEO.

3.4.1 Encoding and decoding schemes

Encoding and decoding schemes are vital to heuristically solve a mathematical model comprising different optimization objectives and constraints (Pasha et al., 2022; Fathollahi-Fard et al., 2019). While the encoding is performed by the SEO-based algorithm (step 2), the random key method (Gonçalves, & Resende, 2011) is used as the decoding scheme applied in steps 3, 5 and 7.

In the proposed DPFSP, the decoding scheme is used for three purposes: (*i*) the selection of operation modes for each machine; (*ii*) the allocation of tasks for each production center and, (*iii*) the scheduling of tasks on the machines of each center. The pseudo-code provided in Figure 3.3 shows how the operating mode is selected for a machine under budget and waste ratio constraints. The matrix received from the main algorithm contains input variables whose values vary between zero and one. The random key method selects the minimum array within this matrix that meets the constraints on budget, waste and operating mode selection (respectively constraints (3.4), (3.5) and (3.10)). If there is no minimum array meeting the 3 constraints, the operating mode with the lowest cost (constraint (3.4)) or the lowest waste ratio (constraint (3.5)), is selected.

```
X; %Input received from the main algorithm
model(); %Input data function
M=model.M; %Number of machines
F=model.F;    %Number of production centers
T=model.T;   %Number of operating modes
JOmtf=model.JOmtf;   %Job opportunities
CJmtf=model.CJmtf;    %Salary of operators
RWmtf=model.RWmtf;  %Ratio of broken products
COmtf=model.COmtf;    %Cost of implementation
B=model.B;       %Budget
MW=model.MW;  %Maximum broken products  ratio
%% Loop for the operating mode selection
Ymtf=zeros(M,T,F);    %Decision variables for operating mode selection
BB=0;     %Counter
MWW=0;    %Counter
XX=X;   %Selection from all the variables
m=1;           %Counter
f=1;           %Counter
while BB<=B && MWW<=MW && m<=M && f<=F
            [a, b]=min(XX(:,m));
            Ymtf(m,b,f)=1;
            BB=BB+Ymtf(m,b,f)*CJmtf(m,b,f)*JOmtf(m,b,f)+ COmtf(m,b,f)*Ymtf(m,b,f);
            MWW=MWW+Ymtf(m,b,f)*RWmtf(m,b,f);
            m=m+1;
            f=f+1;
end
for  m=1:M
     for f=1:F
        if   BB<=B && sum(Ymtf(m,:,f))==0
            [a, b]=min(COmtf(m,:,f));
            Ymtf(m,b,f)=1;
             BB=BB+Ymtf(m,b,f)*CJmtf(m,b,f)*JOmtf(m,b,f)+ COmtf(m,b,f)*Ymtf(m,b,f);
        end
        if   MWW<=MW && sum(Ymtf(m,:,f))==0
            [a, b]=min(COmtf(m,:,f));
            Ymtf(m,b,f)=1;
             MWW=MWW+Ymtf(m,b,f)*RWmtf(m,b,f);
        end
    end
end
Ymtf;
```

Figure 3.3 Pseudo-code describing the allocation of an operating mode to each machine

Arrays generated by the random-key algorithm that contain values between zero and one are sorted to provide the sequence of tasks. The assignment of tasks to production centers and machines is then carried out according to the feasibility of the sequence of tasks with regard to the selected operating mode. Figure 3.4 shows the pseudo-code of these last two decoding phases.

```
Y;        %Input taken from the main algorithm
model(); %Input data function
M=model.M; %Number of machines
F=model.F;    %Number of production centers
T=model.T;   %Number of operating modes
Ymtf;    %Decision variable for operating mode selection
N=model.n;    %Number of tasks
Pnmtf=model.Pnmtf;   %Process time
%% Loop for the task scheduling
Xnimtf=zeros(N, N, M,T,F);    %Decision variables for scheduling
Cimtf=zeros(N,M,T,F);             %Complete time for each position
[a, b]=sort(Y);
for  m=1:M
     for f=1:F
        for t=1:T
            if Ymtf(m,t,f)==1
              for n=1:N
                bb=b(n)
                 Xnimtf(bb,n,m,t,f)=1;
                 Cimtf(n,m,t,f)=Cimtf(n,m,t,f)+ Xnimtf(bb,n,m,t,f)*Pnmtf(bb,m,t,f);
              end
            end
        end
     end
end
Xnimtf;
```

Figure 3.4 Pseudo-code describing the scheduling of tasks

3.4.2 Learning-based SEO

Recently, many modifications and hybridization of heuristic algorithms using learning-based concepts and local search-based operators have been proposed (Zhang et al., 2019; Wang et al., 2020). Like other random-based heuristics, SEO includes exploration and exploitation phases (Fathollahi-Fard et al., 2019). The exploitation phase involves training and retraining of the defenders (Mojtahedi et al., 2021) while social engineering attacks force the exploration of new solutions. The main contribution of this study is the creation of new operators to improve the quality of the non-dominated solutions and to reduce the computation time of the algorithm.

This results in the creation of the Learning-based SEO (LSEO) which dynamically updates the training ratio of the defender (*Alpha*). This automatic updater makes it easier for users to implement the algorithm. After trying a few values for each parameter, the algorithm updates the parameter values as follows:

$$Alpha_{It} = \begin{cases} Lower_{Alpha} + (Upper_{Alpha} - Lower_{Alpha})\left(\frac{It}{MaxIt}\right), \text{ If the new defender dominates the current} \\ Alpha_{It} = Upper_{Alpha} - (Upper_{Alpha} - Lower_{Alpha})\left(\frac{It}{MaxIt}\right), \quad \text{Otherwise} \end{cases} \tag{3.20}$$

where $Alpha_{It}$ is the value of *Alpha* at iteration *It*, $Upper_{Alpha}$ and $Lower_{Alpha}$ are upper and lower bounds of this parameter, respectively, and *MaxIt* is the maximum number of iterations.

This study also proposes a new methodology for carrying out a local search. With each attack, a new defender and attacker are generated using the following formulas:

$$Defender_{new} = \frac{Defender_{old} + Attacker_{old}}{2} + rand \times ((Upper_{bound} - Lower_{bound}) * rand) + Lower_{bound}) \tag{3.21}$$

$$Attacker_{new} = \frac{Defender_{old} + Attacker_{old}}{2} - rand \times ((Upper_{bound} - Lower_{bound}) * rand) + Lower_{bound}) \tag{3.22}$$

where $Defender_{old}$ and $Attacker_{old}$ are respectively the defender and attacker before the attack, while $Defender_{new}$ and $Attacker_{new}$ are respectively the defender and attacker after the attack, and *rand* is a value chosen randomly between zero and one. Furthermore, $Upper_{bound}$ and $Lower_{bound}$ are respectively the upper and lower bounds of this new way of exploring the search area. The proposed multi-objective LSEO algorithm is summarized in Figure 3.5.

```
MaxIt;        %Maximum number of iteration
Nat;          %Number of attacks
Upper_Alpha;        %Maximum training ratio
Lower_Alpha;        %Minimum training ratio
%% Main loop
Create two solutions;
Sort the solutions and select the better one as the attacker;
Another solution is selected as the defender.
t=1;          % Counter
List;         %List of Pareto solutions
while  t≤ MaxIt
     Do the training using equation (3.20).
     nt=1;
     while nt ≤ Nat
           nt=nt+1;
           Select the technique given in equations (3.21) and (3.22) to do an attack;
           Update the defender and the attacker if they can dominate the previous one;
      end
      Exchange the defender and attacker if the defender is able to dominate the attacker;
      Send the attacker to the List;
      Create a new random solution as the defender;
      t=t+1;
end
 Evaluate the List and generate the Pareto fronts;
 Select the non-dominated solutions and show them;
```

Figure 3.5 Pseudo-code of the multi-objective LSEO

3.5 Computational results

In this section, the proposed industrial example and the simulated test studies used to do our analyses are first provided. Then, different criteria and metrics used to assess the algorithms are defined and the parameter values are tuned leading to a fair comparison of the provided algorithms. Next, a validation study is performed to find the exact solution using an epsilon-constraint method. Then, the performance of our algorithms is compared to that of various traditional and recent algorithms using evaluation metrics. The robustness of the proposed optimization model is also evaluated by a sensitivity analysis. It should be noted that our codes were written in GAMS and MATLAB software and implemented on a laptop with 1.7 GB CPU and 6.0 GB RAM.

3.5.1 Industrial example and tests

Canadian Wood Products (CWP)[3] is one of the well-known practitioners of the wood industry in Canada. This company is the leader in the production and distribution wood products in North America. It has three main products, including softwoods, industrial and architectural lumbers. For each, a specific operation mode must be installed on the machines. Moreover, six operational tasks are required including cutting, custom processing, drying, classifying, storing and loading. Last but not least, the CWP has three main production and distribution centers in Buffalo, Montreal and Concord.

The industrial example of the company CWP is used to show the applicability of the optimization method developed. For security reasons, the actual values of the parameters are not accessible and therefore, estimated values are provided. Moreover, in order to evaluate our approach with 3 levels of model complexity, 12 tests were created, 4 tests for each level of complexity, namely, small, medium and large models as shown in Table 3.3.

Table 3.3 Test studies used to evaluate the proposed algorithm

Complexity level of the model	Number of test studies	Number of centers (F)	Number of machines (M)	Number of operating modes (T)	Number of tasks (N)
Small Size	Industrial example	3	3	3	6
	T1	2	2	2	4
	T2	2	2	2	8
	T3	2	4	2	20
	T4	3	4	3	30
Medium Size	T5	3	6	2	30
	T6	3	6	3	40
	T7	4	8	4	30
	T8	4	8	5	40
Large Size	T9	6	12	4	80
	T10	6	12	5	100
	T11	8	16	6	80
	T12	10	16	6	100

[3] https://canadianwood.ca/

Since the optimization model proposed for a sustainable DPFSP is novel depending on the different production centers and operating modes as well as social factors, there is no benchmark dataset available corresponding to our optimization model. In this regard, the possible value ranges for the model parameters are presented in Table 3.4. To fix the parameters values, we run random functions for each test size and then save the values.

Table 3.4 Ranges of values for model's parameters

Parameter	**Range**
P_{nmtf}	*randi*([2, 8],N, M, T, F)
CO_{mtf}	*randi*([8, 20],M, T, F)*10^4
JO_{mtf}	*randi*([2, 9],M, T, F)
CJ_{mtf}	*randi*([8, 20],M, T, F)
LD_{mtf}	*randi*([8, 30],M, T, F)
WJ	0.9
WL	0.1
RW_{mtf}	*rand*(M, T, F)*0.1
IEC_{mtf}	(*randi*([8, 12],M, T, F)+*rand*())*10^5
UEC_{mtf}	(*randi*([2, 7],M, T, F)+*rand*())*10^5
EC_{mtf}	(*randi*([20, 40],M, T, F)+*rand*())*10^5
B	*randi*([*round*(*sum*(JO_{mtf}.* CJ_{mtf} + CO_{mtf})/2), *round*(*sum*(JO_{mtf}.* CJ_{mtf} + CO_{mtf}))
MW	**if** *sum*(RW_{mtf})>1 *randi*([*round*(*sum*(RW_{mtf})/2), *round*(*sum*(RW_{mtf})) **else** *rand*()+(*sum*(RW_{mtf})/2) **end**

**randi* is a function which generates random integer numbers between lower and upper bounds.
**rand* is a function which generates random continuous numbers between zero and one.
**round* is a function which transforms a continuous number to the closest integer number.
**sum* is a function to sum numbers contained in a matrix.

3.5.2 Assessment metrics and parameters tuning

As mentioned earlier, this study develops LSEO as an improvement to SEO. This study not only compares the performance of LSEO with that of SEO but also with the performance of other well-known and state-of-the-art algorithms in the literature. In this regard, the Non-dominated Sorting Genetic Algorithm (NSGA-II) (Deb et al., 2000), the enhanced Strength of Pareto Evolutionary Algorithm (SPEA2) (Zitzler et al., 2001), the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) (Zhang et al., 2007) as well as two recent algorithms comprising the Multi-Objective Brain Storm Optimization (MOBSO) (Fu et al., 2019) and the Multi-Objective Keshtel Algorithm (MOKA) (Cheraghalipour et al., 2018) are used.

These algorithms are evaluated using assessment metrics. In addition to the algorithm computation time criterion (CPU time), the Number of Pareto Solutions (NPS) (Zitzler, 1999), the Mean Ideal Distance (MID) (Zitzler et al., 2001), the Maximum Spread (MS) (Zitzler et al., 2003) and the Hypervolume (HV) (Zitzler et al., 2007) are considered to evaluate the Pareto solutions found by the algorithms. These metrics are defined hereafter:

- NPS is the number of non-dominated solutions in the Pareto optimal set. A higher value of this metric shows a better diversity of the solutions (Zitzler, 1999).
- MS measures the distance between the best and the worst solutions in the optimal Pareto set. It can be formulated as follows:

$$MS = \sqrt{\left(\sum_{j=1}^{NO} (Z_j^{Max} - Z_j^{Min})\right)^2} \tag{3.23}$$

where Z_j^{Max} and Z_j^{Min} are respectively the maximum and the minimum value of the objective j among all the solutions Along with the NPS metric, this metric evaluates the diversity of the solutions. A higher value of the MS metric means a better capability of the algorithm (Zitzler et al., 2003) to find an optimal solution.

- MID measures the distance between solutions in the Pareto optimal set and we can formulate it as follows:

$$MID = \frac{\sum_{i=1}^{NPS} \sqrt{\left(\sum_{j=1}^{NO} \frac{Z_j^i - Z_j^{Best}}{Z_j^{Max} - Z_j^{Min}}\right)^2}}{NPS} \tag{3.24}$$

where *NO* is the number of objectives, Z_j^i is the solution i for objective j, and Z_j^{Best} is the maximum or minimum value with regards to the type of the objective function. A lower value of this metric shows a faster convergence of the solution (Zitzler et al., 2001).

- HV computes the space of non-dominated solutions. It is difficult to calculate HV exactly as it cannot be formulated mathematically. An approximation method such as Monte Carlo is usually used to compute this metric. In this study, the simulation method of Zitzler et al., (2007) was used to quantify HV. A higher value of this metric shows a better performance of the Pareto set.

With regard to the above-mentioned criteria, the metaheuristic algorithms must be tuned before the validation and comparison studies. Good tuning helps algorithms achieve their best performance and therefore, leads to a fair comparison (Fathollahi-Fard et al., 2019). Consequently, in this study, the parameters were tuned using the Taguchi method (Roy, 2010). Taguchi first uses orthogonal arrays to reduce the number of experiments using only selected experiments. For example, if an algorithm has five parameters and each has three candidate values, the total number of experiments for one run is 3^5 =243. However, Taguchi uses an orthogonal array of L_{27} reducing the number of experiments to 27. Taguchi is also based on noise and control factors which are evaluated by the Signal to Noise (S/N) ratio and Relative Percentage Deviation (RPD), respectively. In the context of multi-objective optimization, assessment metrics are used. The S/N ratio can be formulated as:

$$S/N = -10 \times \log_{10}\left(\frac{\sum_i {HV_i}^2}{n}\right) \tag{3.25}$$

where n is the number of orthogonal arrays and HV_i is the response value of the i^{th} orthogonal array. Similar to the *HV* metric, a higher value of S/N is preferable for this noise factor. As shown in the following formula, the control factor evaluated by RPD includes *MID* and *MS* metrics that quantify the precision and diversity of Pareto solutions, respectively:

$$RPD = \frac{MID}{MS} \tag{3.26}$$

Thus, a lower value of *RPD* means better performance of the algorithm.

In order to make an unbiased comparison, the maximum number of fitness evaluations is set to the same values for all algorithms under evaluation based on the size of the model's complexity levels: 25000, 50000 and 100000 for small, medium and large sizes, respectively. Therefore, for SEO and LSEO as one-solution algorithms, the maximum number of iterations (*MaxIt*) and maximum number of attacks (*Nat*) are set to 500 and 50 respectively for small sizes (500×50 = 25000), to 1000 and 50 for medium sizes (1000×50 = 50000) and up to 2000 and 50 for large sizes (2000×50 = 100000). For population-based algorithms including NSGA-II, SPEA2, MOEA/D, MOBSO and MOKA, the maximum number of generations (*MaxIt*) and population size (*nPop*) are respectively set to 250 and 100 for small sizes (250×100 = 25000), to 500 and 100 for medium sizes (500×100 = 50000) and to 1000 and 100 for large sizes (1000×100 = 100000). Other parameters of each algorithm were adjusted based on candidate values identified in previous studies (Cheraghalipour et al., 2018; Fathollahi-Fard et al., 2018; Mojtahedi et al., 2021; Fu et al., 2019) as reported in Table 3.5.

Table 3.5 Candidate values for parameters of algorithms under evaluation

Algorithms	Parameters	Candidate values		
SEO	Percentage of training ($Alpha$)	0.1	0.3	0.5
	Rate of attack (*Betta*)	0.05	0.15	0.25
LSEO	Upper bound of $Alpha$ ($Upper_{Alpha}$)	0.8	0.9	1
	Lower bound of $Alpha$ ($Lower_{Alpha}$)	0	0.1	0.2
NSGA-II	Percentage of crossover (P_C)	0.5	0.6	0.7
	Percentage of mutation (P_m)	0.1	0.15	0.2
SPEA2	Number of archive (N_A)	50	75	100
MOEA/D	Number of subproblems considered in MOEA/D (*N*)	150	200	250
	Number of weight factors (*T*)	12	25	50
MOBSO	Probability of each generation (P_g)	0.2	0.4	0.8
	Probability of first cluster (P_{c1})	0.2	0.4	0.6
	Probability of second cluster (P_{c2})	0.2	0.4	0.6
MOKA	Number of swirling (*NS*)	2	3	5
	Percentage of lucky Keshtels (*N1*)	0.1	0.2	0.4
	Percentage of moving Keshtels (*N2*)	0.4	0.5	0.6
	Percentage of random Keshtels (*N3*)	$N3 = 1 - N1 - N2;$		

The orthogonal array for SEO, LSEO, NSGA-II and MOEA/D is a full factorial method (3×3 = 9). As such, SPEA2 has three tests. The orthogonal array of L_9 is used for MOBSO and MOKA. Based on the calculation of the noise and the control factors, the best candidate value for each parameter is reported in Table 3.6.

Table 3.6 Tuned parameters of the algorithms

Algorithm	Parameters
SEO	$Alpha$=0.3; $Beta$=0.05;
LSEO	$Upper_{Alpha} = 1;\ Lower_{Alpha} = 0.1;$
NSGA-II	$P_C = 0.7;\ P_m = 0.1;$
SPEA2	$N_A = 100;$
MOEA/D	$N = 200; T = 50;$
MOBSO	$P_g = 0.8; P_{c1} = 0.4;\ P_{c2} = 0.2;$
MOKA	$NS = 3; N1 = 0.2; N2 = 0.5;\ N3 = 0.3;$

3.5.3 Validation

The Epsilon-Constraint (EC) method (Haimes et al., 1971) is solely used to find exact solutions to our example of an industrial problem in order to validate the performance of the proposed algorithms. This algorithm optimizes one main objective and uses upper and lower bounds for other objective functions. As the economic criterion is generally more important than environmental and social criteria for production managers, the first objective is chosen in this study as the main objective. Therefore, the problem addressed by the EC method can be formulated as follows:

$$
\begin{gathered}
\min_{Economic} Z_1 \\
s.t.\ \text{Constraints (3.4) to (3.19)} \\
Z_2 \leq EC_2 \\
Z_3 \geq EC_3 \\
Z_2^{Min} \leq EC_2 \leq Z_2^{Max} \\
Z_3^{Min} \leq EC_3 \leq Z_3^{Max}
\end{gathered}
\tag{3.27}
$$

where EC_2 and EC_3 are allowable bounds of the second and third objectives, respectively, while the lower and upper bounds for the second objective are $Z_2^{Min}$and Z_2^{Max}, respectively. As such, Z_3^{Min} and Z_3^{Max} are the lower and upper bounds of the third objective function, respectively. To find these bounds, we solve the model separately for each objective function using the epsilon constraint method. If only the makespan criterion is optimized, the objective values are $Z_1^* = 83, Z_2 = 1.60E + 08$ and $Z_3 = 22.9$. The CPU time for this run is 4.38 seconds. If only the environmental criteria are minimized, the objective values are $Z_1 = 90, Z_2^* = 1.26E + 08$ and $Z_3 = 23.9$. The CPU time for this run is 4.57 seconds. Finally, if the social criteria are maximized, the objectives are $Z_1 = 90, Z_2 = 1.51E + 08$ and $Z_3^* = 27.9$. The CPU time for this run is 4.27 seconds. Therefore, the lower and upper bounds of the second objective are set to $Z_2^{Min} = 1.26E + 08$ and $Z_2^{Max} = 1.60E + 08$, respectively. Similarly, the lower and upper bounds of the third objective are respectively $Z_3^{Min} = 22.9$ and $Z_3^{Max} = 27.9$.

To generate more Pareto solutions, the average of upper and lower bounds of the objectives is considered. However, there is no feasible solution when this average value is used. At the end, the total time to run the EC method to solve our industrial test, is 13.22 seconds.

All the Pareto solutions found by EC, SEO and LSEO are reported in Table 3.7. These solutions are depicted in Figure 3.6. One disadvantage of the EC method is that it is limited in its capacity to generate many Pareto solutions. However, SEO and LSEO are able to create 12 and 16 solutions, respectively. The results shown in Table 3.7 and Figure 3.6 confirm that SEO and LSEO are able to create high quality solutions like EC does.

Table 3.7 Pareto solutions after solving the industrial example

EC			SEO			LSEO		
Z_1	Z_2	Z_3	Z_1	Z_2	Z_3	Z_1	Z_2	Z_3
83	1.6×10^8	22.9	85	1.85×10^8	21.2	84	1.85×10^8	20.9
90	1.26×10^8	23.9	85	1.82×10^8	21.4	84	1.83×10^8	21.2
90	1.51×10^8	27.9	86	1.80×10^8	21.8	84	1.80×10^8	21.8
-	-	-	87	1.78×10^8	22.2	86	1.74×10^8	22.5
-	-	-	87	1.76×10^8	22.4	86	1.66×10^8	23.2
-	-	-	88	1.72×10^8	22.8	86	1.64×10^8	23.6
-	-	-	88	1.66×10^8	23.2	86	1.58×10^8	24.2
-	-	-	88	1.64×10^8	23.6	87	1.54×10^8	24.6
-	-	-	89	1.58×10^8	23.8	87	1.52×10^8	25.2
-	-	-	89	1.54×10^8	24.6	87	1.51×10^8	25.5
-	-	-	89	1.52×10^8	25.2	87	1.50×10^8	25.8
-	-	-	90	1.49×10^8	26.7	88	1.50×10^8	26.2
-	-	-	-	-	-	88	1.49×10^8	26.7
-	-	-	-	-	-	88	1.46×10^8	26.9
-	-	-	-	-	-	88	1.45×10^8	27.4
-	-	-	-	-	-	88	1.44×10^8	27.6

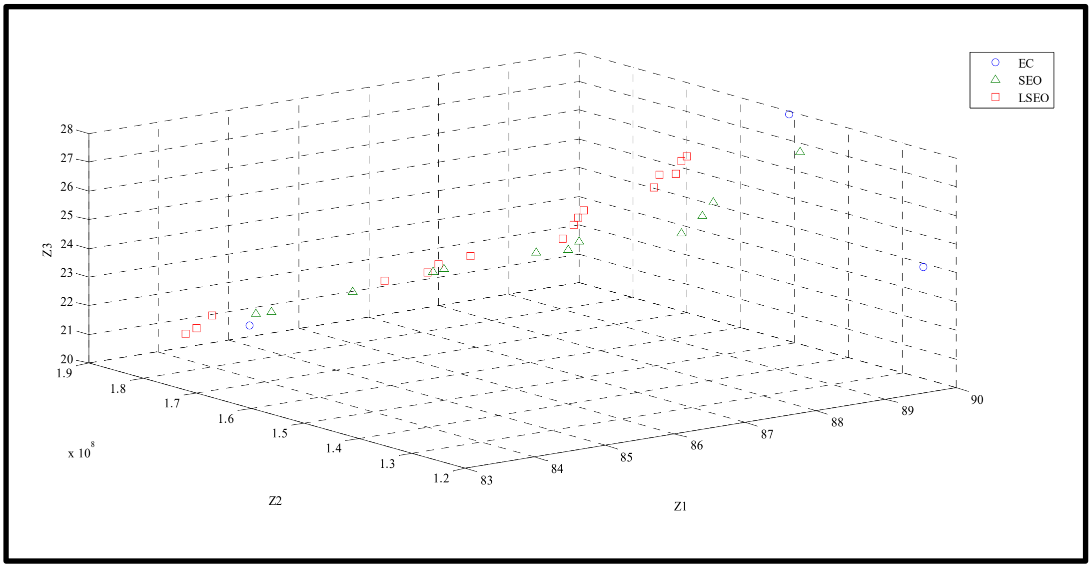
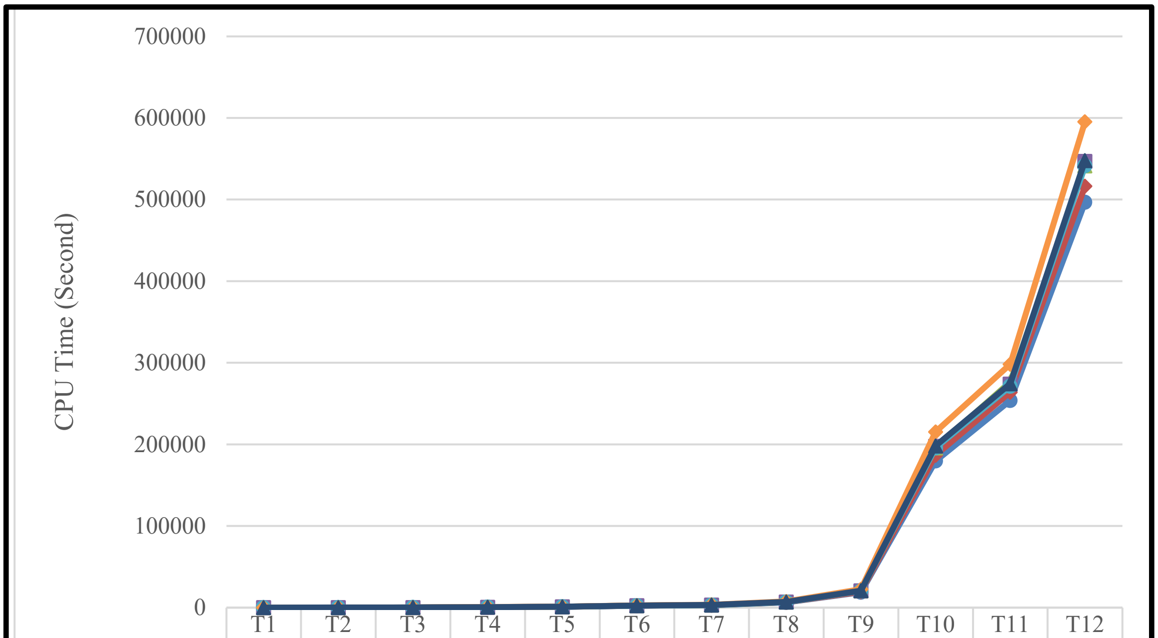


Figure 3.6 Pareto solutions for CWP company

3.5.4 Comparison

To show the high performance of the proposed LSEO, it has been compared to its original version of SEO as well as state-of-the-art methods like NSGA-II, SPEA2, and MOEA/D and two recent algorithms including MOBSO and MOKA. In this regard, 12 test problems with different complexity levels are solved by these algorithms. Due to the randomization of these algorithms, we run them thirty times and the average of their results is considered reliable.

The first criterion used in this comparison is the CPU time. This criterion also confirms the level of complexity of the test studies. Figure 3.7 shows the CPU times required by the algorithms to solve the simulated test studies. As can be seen, LSEO and SEO are faster than other algorithms. However, as can be seen from this chart, the CPU times of the algorithms are of the same order of magnitude. This is because the number of fitness evaluations is considered the same for all algorithms. From these results, SEO and MOBSO methods are identified as the fastest and slowest in the majority of the simulated test studies, respectively.



	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
SEO	15,163	21,412	99,729	437,53	871,81	2228,7	2919,4	6320	18770	179625	253760	496634
LSEO	15,315	21,626	100,73	441,91	897,96	2295,5	3007	6509,6	19333	186810	263911	516499
NSGA-II	18,196	25,694	119,67	525,04	1020	2407	3153	6825,6	20460	195792	276599	541331
SPEA2	17,892	25,266	117,68	516,29	1009,8	2382,9	3184,5	6893,8	20664	197750	273833	546744
MOEA/D	20,47	28,906	134,63	590,67	999,72	2359,1	3152,7	6824,9	20458	195772	271095	541277
MOBSO	22,517	31,797	148,1	649,74	1099,7	2595	3467,9	7507,4	22503	215349	298204	595404
MOKA	20,716	29,253	136,25	597,76	1011,7	2387,4	3190,5	6906,8	20703	198121	274348	547772

Figure 3.7 CPU times of the algorithms for solving the simulated test studies

Four multi-objective criteria including NPS, MID, MS and HV are considered to evaluate the quality of Pareto solutions found by the algorithms. Their results are respectively reported in Table 3.8, 3.9, 3.10, and 3.11. In these tables, the best values are highlighted in bold.

Table 3.8 Evaluation of algorithm performance using the NPS metric

Test problem	SEO	LSEO	NSGA-II	SPEA2	MOEA/D	MOBSO	MOKA
T1	**14**	8	10	8	6	4	3
T2	20	**34**	26	16	15	8	12
T3	64	55	**76**	39	24	28	33
T4	23	**29**	16	8	19	18	15
T5	44	**62**	55	39	44	21	27
T6	76	78	**88**	56	74	36	25
T7	88	**102**	100	75	66	49	52
T8	105	**116**	100	79	81	68	42
T9	**215**	148	100	100	97	73	75
T10	**309**	188	100	100	100	93	56
T11	118	**172**	100	95	92	82	96
T12	136	**211**	100	100	100	100	88

Table 3.9 Evaluation of algorithms performance using the MID metric

Test problem	SEO	LSEO	NSGA-II	SPEA2	MOEA/D	MOBSO	MOKA
T1	39.6	40.8	33.2	29.8	37.5	26.5	**24.5**
T2	45.7	56.4	**29.8**	33.4	45.6	37.9	45.4
T3	102.6	88.7	78.6	92.7	**67.5**	85.2	86.5
T4	115.4	**95.4**	102.6	122.6	109.5	142.6	108.5
T5	276.3	188.7	**96.5**	112.5	119.6	189.4	242.5
T6	197.5	186.5	254.3	297.3	109.6	**98.3**	144.2
T7	**149.2**	156.2	188.7	206.3	188.5	193.2	174.5
T8	**228.4**	256.3	345.2	305.2	288.1	306.5	283.5
T9	319.5	**252.6**	428.9	402.4	377.5	392.6	275.1
T10	177.6	**144.3**	388.1	265.1	244.2	271.9	193.2
T11	188.9	**156.2**	504.2	209.3	235.1	228.3	298.2
T12	244.3	219.5	399.1	275.1	218.5	275.3	**199.4**

Table 3.10 Evaluation of algorithms performance using the MS metric

Test problem	SEO	LSEO	NSGA-II	SPEA2	MOEA/D	MOBSO	MOKA
T1	6984999.6	5894376.2	**7068555.1**	4982399.4	5068822.3	3894506.3	5884382.5
T2	6985734.5	**8648332.6**	5068429.5	2285694.1	6093392.5	8260555.1	7489302.5
T3	8443506.2	7085439.6	6089427.4	7053277.3	3885467.2	**8543772.8**	7094463.2
T4	**9956309.4**	8544288.3	7095275.2	4096855.3	4035588.2	3095668.2	6435068.2
T5	**10753982.3**	9885063.5	7068329.5	7047783.2	8490355.2	6988543.2	8665447.2
T6	8975664.2	**10546783.2**	8946684.5	8996583.2	6047752.5	7864733.5	9627543.5
T7	12527709.2	10546782.2	1078822.5	**12780423.2**	10247685.2	9987402.4	9902545.2
T8	**12563902.7**	12036547.3	11664893.4	10522910.3	9654553.6	10454893.2	11784405.2
T9	**13829504.2**	10863892.5	10573902.6	11829044.6	8562981.5	9924893.6	10452855.3
T10	12872895.3	11678649.3	11539671.5	10997783.5	10782861.5	11653782.5	**12994673.9**
T11	11673785.6	**12982901.5**	12738594.5	12452895.1	11770839.6	12620256.4	10097855.3
T12	13678864.7	**14014582.6**	12922099.1	12672895.1	11852856.4	10451197.5	10735784.6

Table 3.11 Evaluation of algorithms performance using the HV metric

Test problem	SEO	LSEO	NSGA-II	SPEA2	MOEA/D	MOBSO	MOKA
T1	2.87E+09	**3.83E+09**	1.98E+09	3.72E+09	1.09E+09	1.54E+09	2.65E+09
T2	4.74E+09	5.53E+09	3.67E+09	**7.73E+09**	2.87E+09	1.93E+09	3.54E+09
T3	4.38E+09	**5.25E+09**	3.87E+09	4.29E+09	3.28E+09	2.99E+09	3.71E+09
T4	6.39E+09	7.8E+09	**8.54E+09**	6.85E+09	5.78E+09	4.18E+09	7.39E+09
T5	3.86E+09	**8.88E+09**	6.78E+09	7.38E+09	5.68E+09	7.58E+09	6.98E+09
T6	7.63E+09	7.98E+09	7.58E+09	6.58E+09	**9.38E+09**	6.18E+09	8.28E+09
T7	9.23E+09	**9.98E+09**	7.48E+09	8.38E+09	7.48E+09	8.38E+09	5.98E+09
T8	**1.86E+10**	1.15E+10	9.54E+09	1.04E+10	9.73E+09	1.82E+10	9.37E+09
T9	2.75E+10	1.86E+10	1.45E+10	2.67E+10	1.86E+10	**2.99E+10**	1.09E+10
T10	3.54E+10	**3.97E+10**	2.87E+10	1.87E+10	2.07E+10	2.18E+10	2.06E+10
T11	3.87E+10	2.58E+10	1.96E+10	**5.84E+10**	4.38E+10	2.97E+10	3.6E+10
T12	**6.85E+10**	3.29E+10	3.65E+10	4.87E+10	6.64E+10	3.79E+10	4.19E+10

To find the best algorithm, the Relative Deviation Index (RDI) (Rothe et al., 1996) is used:

$$RDI = \frac{|Best_{Metric} - Alg_{Sol}|}{Max_{Sol} - Min_{Sol}} \tag{3.28}$$

where Max_{Sol} and Min_{Sol} are the maximum and minimum values for each metric, respectively. Alg_{Sol} is the value of a metric for a specific algorithm while $Best_{Metric}$ is the best value of the metric obtained among all the algorithms. It goes without saying that a lower value of RDI is more preferable.

After transforming the metrics based on the RDI, the interval plot for each metric is depicted statistically in Figure 3.8. Based on the NPS metric criterion (Figure 3.8(a)), the developed LSEO shows the best performance followed by the SEO algorithm. However, MOBSO and MOKA perform very poorly on this metric. Regarding the MID metric criterion (Figure 3.8(b)), again, the LSEO algorithm shows the best performance. MOKA is better than other algorithms in this case. The MOEA/D is also good on this criterion and better than the SEO algorithm. However, NSGA-II is the worst algorithm in this metric. Based on the MS metric (Figure 3.8(c)), SEO and LSEO algorithms are clearly better than other algorithms. For the HV metric (Figure 3.8(d)), the same conclusion is drawn from the results. In conclusion, as can be seen from the interval plots, the proposed LSEO algorithm achieves the best performance in this comparative study.

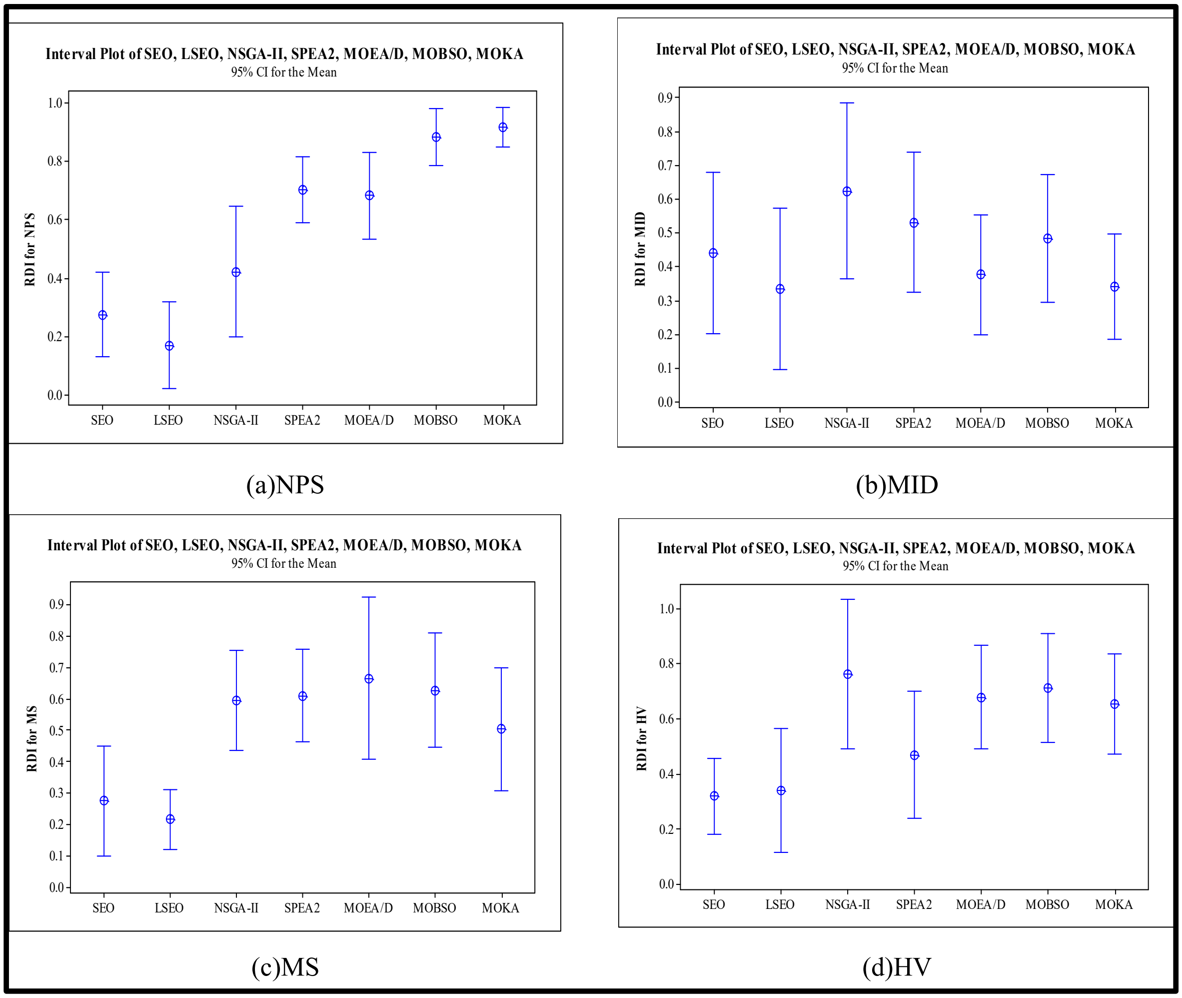


Figure 3.8 Interval plots based on RDI analyzing the algorithms performance

3.5.5 Sensitivity analyses

To evaluate the robustness of the optimization model developed, some sensitivity studies are carried out here. First, the Pareto solutions found by LSEO to be the best algorithm in this study, are sorted by the ideal distance criterion. Then, the first solution of this Pareto set is noted in Table 3.12. The variations of the objectives in this table are shown in Figure 3.9.

Table 3.12 The first solution from the set of sorted Pareto solutions from LSEO

Test problem	Z_1	Z_2	Z_3
T1	39	3,30E+07	10
T2	76	7,78E+07	10,9
T3	399	4,07E+08	19,6
T4	631	7,87E+08	41,2
T5	909	1,27E+09	65,9
T6	1236	1,89E+09	47,9
T7	1146	2,40E+09	78,9
T8	1655	3,16E+09	117,9
T9	4885	1,54E+10	208,9
T10	6141	1,79E+10	192,1
T11	6446	2,85E+10	355,8
T12	8177	3,76E+10	487,1

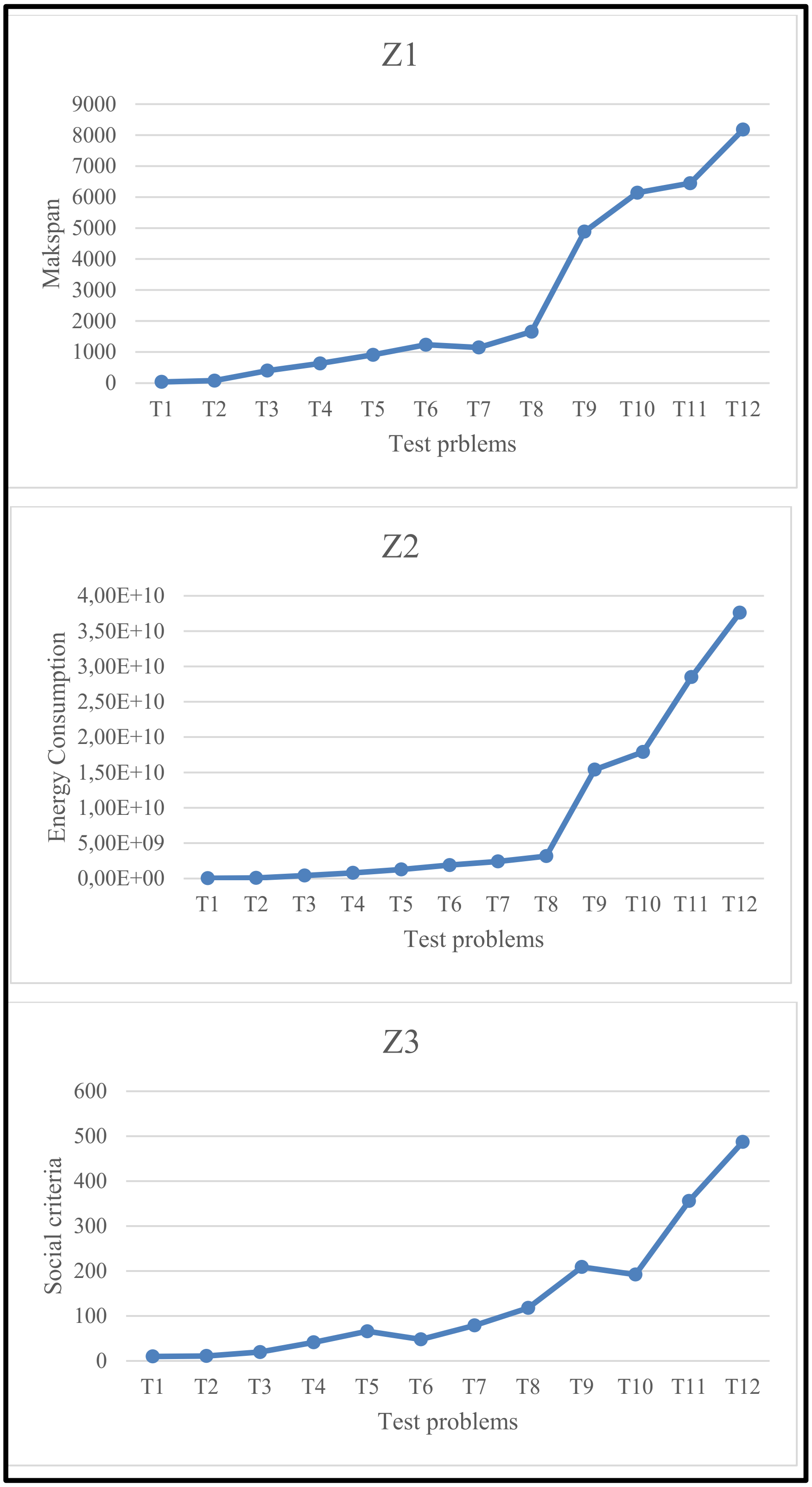


Figure 3.9 Variations of the objective values

To show the impact of the parameters of the model developed for the decision-makers of the CWP company as a leader in the wood industry in Canada, the EC method is selected to solve the industrial example in our sensitivity analyses. Of these parameters, four important ones are selected for modification. The company's budget (B), the maximum waste (MW) as well as the social weights for job opportunities (WJ) and lost working days (WL) are considered for our analyses. For each parameter, certain modifications are done by four scenarios: S1 to S4 and the values of objective functions in each scenario are indicated. Note that in our sensitivity analyses, in addition to makespan, the total flow-time ($\sum_{f=1}^{F} CT_f$) is also considered to better show the impact of parameters on the economic criteria.

The sensitivity analysis of the company's budget is reported in Table 3.13. In four scenarios, the company's budget goes from 2,594,561$ (S1) to 2,000,000$ (S4). The values of the objective functions for each case are noted. As can be seen, there is no change in the makespan criterion, considered as the first objective while an increase in the total flow-time is observed in the last scenario. This means that reducing the budget to be less than 2,300,000$ has a negative economic impact. Although the values of the second and third objectives have some variations, they have been increased if the first scenario is compared to the last scenario. The behavior of these objectives (except makespan) is drawn in Figure 3.10.

Table 3.13 Sensitivity analysis on the budget of company

Scenarios	Value of the company budget ($)	Z_1 (Makespan)	Z_1 (Flow-time)	Z_2	Z_3
S1	2594561	83	231	1.60E+08	22.9
S2	2400000	83	231	1.39E+08	24.2
S3	2300000	83	231	1.51E+08	25
S4	2000000	83	235	1.73E+08	24.2

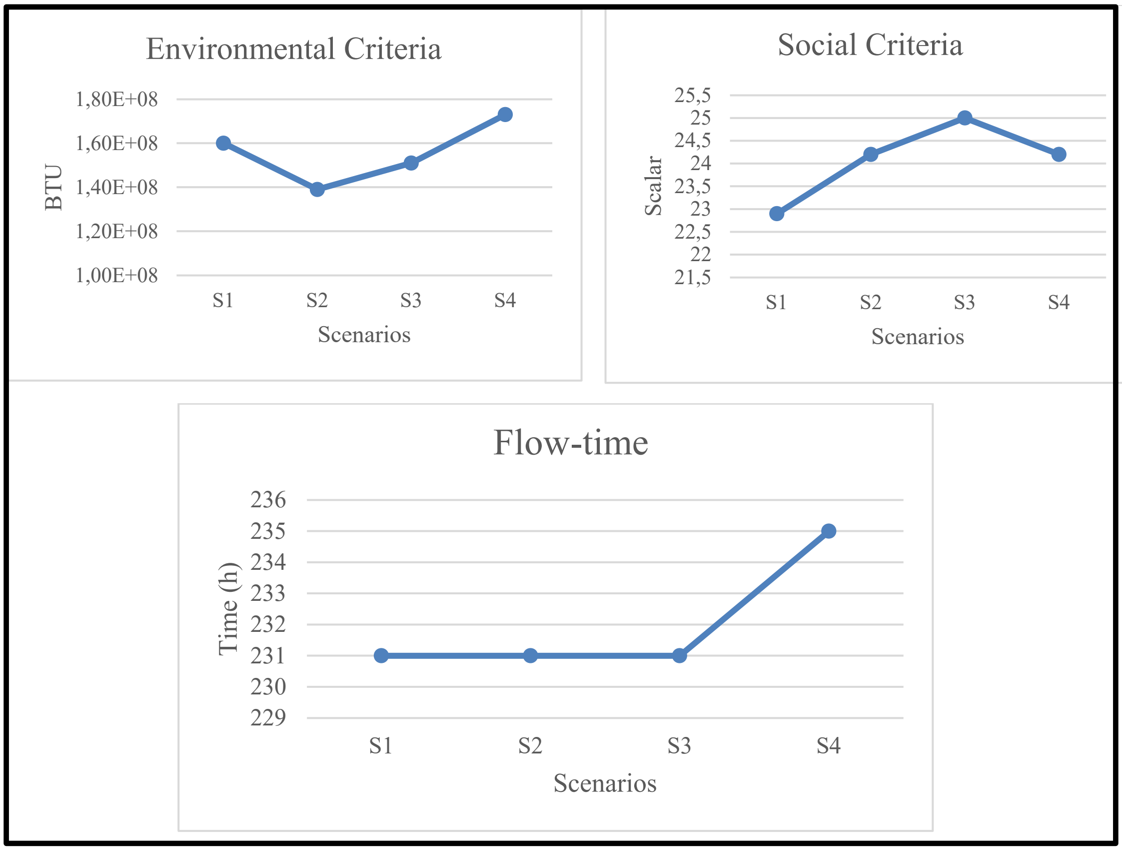


Figure 3.10 Sensitivity analysis on the budget of company

As indicated in Table 3.14, the sensitivity analysis is performed on the value of the maximum waste products. We have reduced the maximum waste products from 2.5 units to 1 unit. This parameter only has an impact on the values of the environmental criteria. The economic criteria including both makespan and total flow-time as well as the third objective functions have not changed. Decreasing the maximum amount of waste products leads to an increase in energy consumption as a second objective. This behavior is illustrated in Figure 3.11.

Table 3.14 Sensitivity analysis on the maximum waste products

Number of Scenario	Value of maximum waste products	Z_1 (Makespan)	Z_1 (Flow-time)	Z_2	Z_3
S1	2.5	83	231	1.53E+08	22.9
S2	2	83	231	1.60E+08	22.9
S3	1.5	83	231	1.71E+08	22.9
S4	1	83	231	1.74e+08	22.9

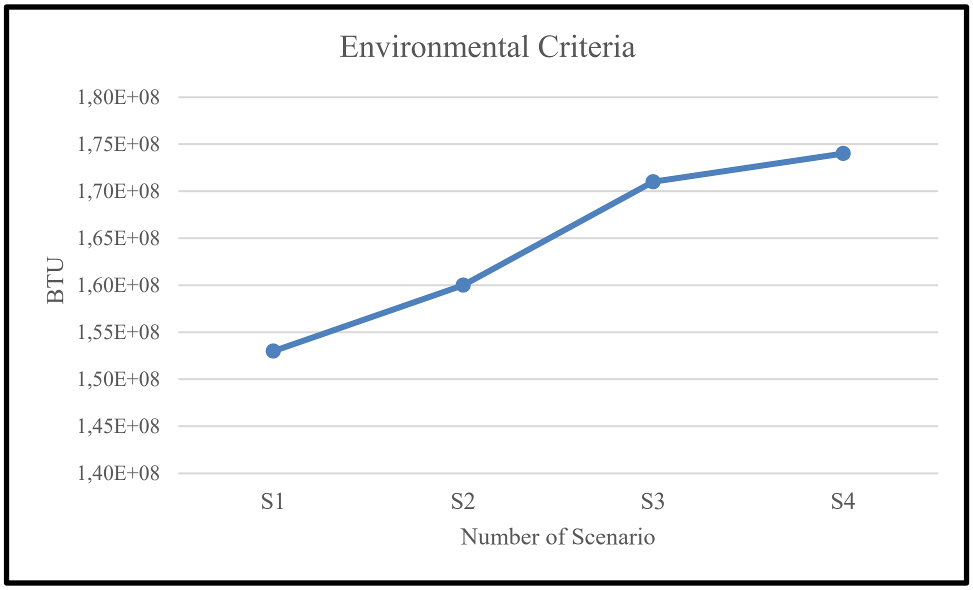


Figure 3.11 Sensitivity analysis on the maximum waste products

The final sensitivity analysis looks at the value of social weights for job opportunities and lost working days as the third objective function as shown in Table 3.15. To better show the impact of changes on these two parameters, the main objective of the EC is changed from makespan to social criteria as the third objective function. In the four scenarios, we reduced the impact of job opportunities while increasing the impact of lost working days. Except for makespan, other criteria show changes. These behaviors are illustrated in Figure 3.12. It shows that the total flow-time decreases in these scenarios except in S2. The energy consumption shows an increase if the first scenario is compared to the last scenario. However, in the S3 scenario, it shows a reduction. Finally, the social objective shows a strong reduction in all scenarios.

Table 3.15. Sensitivity analysis on the social weights

Number of Scenario	Social weights	Z_1 (Makespan)	Z_1 (Flow-time)	Z_2	Z_3
S1	*WJ*=0.99; *WL*=0.01;	90	238	1.48E+08	46.89
S2	*WJ*=0.9; *WL*=0.1;	90	238	1.51E+08	27.9
S3	*WJ*=0.8; *WL*=0.2;	90	235	1.46E+08	7.2
S4	*WJ*=0.7; *WL*=0.3;	90	233	1.65E+08	-24.9

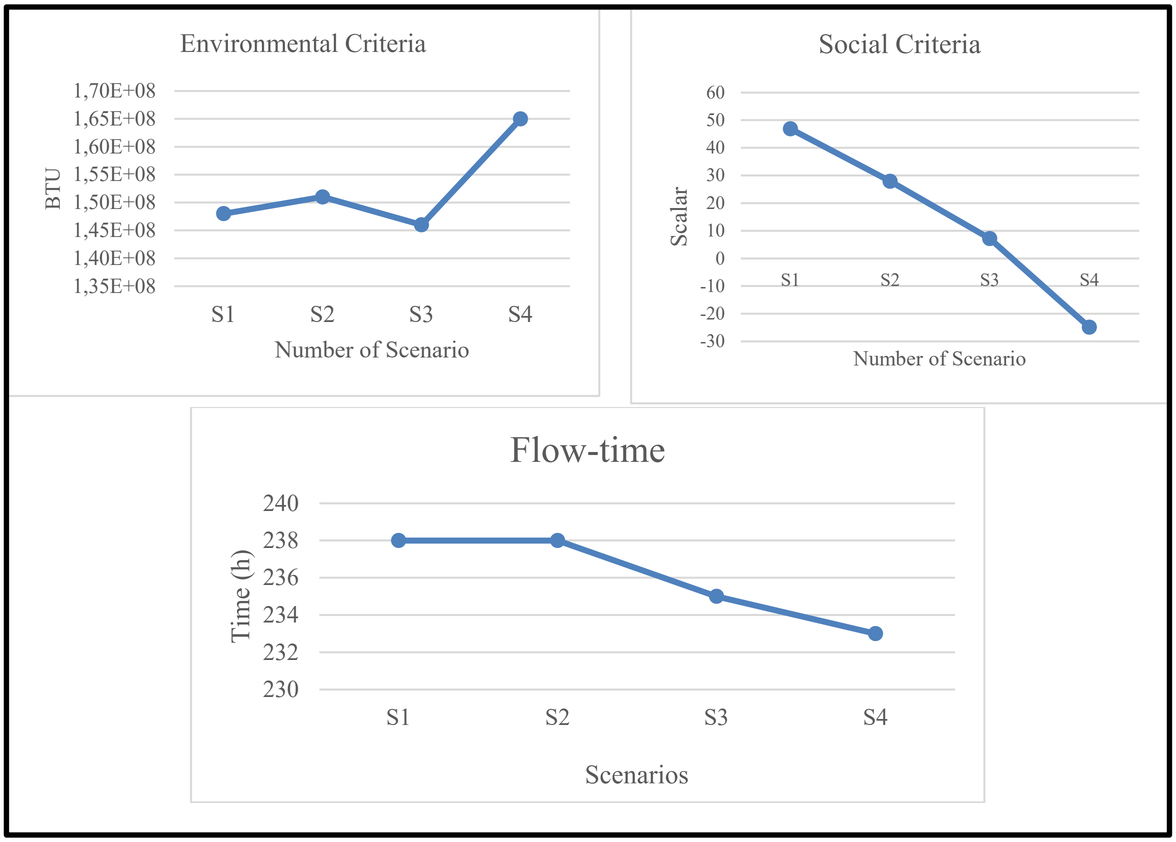


Figure 3.12 Sensitivity analysis on the social weights

3.6 Discussions, and managerial insights

Incorporating sustainability into manufacturing processes is vital for robust competitiveness and societal well-being. This study's development of a sustainable DPFSP model with a focus on all the economic, environmental, and social factors represents a significant stride in this direction. It underscores the importance of integrating sustainability goals into production scheduling strategies.

A unique aspect of this study is its consideration of social sustainability factors, specifically job opportunities and lost working days, within the DPFSP context. By highlighting the need to balance economic and social objectives, this research addresses a critical gap in the literature. For manufacturing managers, this insight signals the importance of not only optimizing production efficiency but also nurturing a workforce and community well-being.

The introduction of a multi-objective learning-based SEO algorithm is another noteworthy contribution. It outperforms other optimization methods, including the epsilon constraint method and established multi-objective metaheuristic algorithms. This finding suggests that advanced optimization techniques can significantly enhance decision-making in production scheduling, offering manufacturing managers a powerful tool for achieving sustainability objectives.

Performance comparison using multiple multi-objective metrics demonstrates the consistent superiority of the LSEO algorithm over other approaches. These metrics, such as NPS, MID, MS, and HV, provide manufacturing managers with a clear framework for assessing optimization methods based on their specific sustainability goals. This insight empowers decision-makers to select the most effective approach for their unique context.

The sensitivity analysis conducted on the company's budget and environmental parameters underscores the importance of adaptability in production scheduling. Budget constraints can impact economic objectives, emphasizing the need for sufficient resource allocation to maintain efficient processes. Moreover, changes in maximum waste levels primarily affect environmental criteria, indicating that reducing waste can lead to lower energy consumption, aligning with sustainability targets.

Adjusting the social weights for job opportunities and lost working days highlights the trade-offs between economic and social objectives. This dynamic approach to balancing these factors necessitates careful consideration and strategic decision-making. Overall, this study equips manufacturing managers with insights and tools to optimize production schedules while achieving sustainability goals, emphasizing the interconnectedness of economic, environmental, and social dimensions in modern production systems. It also provides a foundation for future research directions, offering opportunities to explore social and environmental aspects further and refine optimization approaches for real-world manufacturing challenges.

3.7 Conclusions, recommendations, and future works

The DPFSP traditionally aims to minimize makespan or total flow time based on economic criteria. However, based on the concept of TBL, traditional modeling of the DPFSP is not able to simultaneously cover all economic, environmental and social criteria. This study developed a sustainable DPFSP with the assumption of different production centers and operating modes on machines that have a strong impact on environmental and social criteria. This study which considers job opportunities and lost working days as social factors is the first study in the area of DPFSP. Therefore, a multi-objective optimization model was developed to approach a sustainable TBL-based DPFSP.

One idea of this paper was to virtually meet the challenge of sustainable development based on the TBL concept for wood production in Canada. In this regard, CWP has been selected as a full-scale application for our optimization model. Having different simulated test studies to analyze the complexity of this NP-hard model, this study proposed a multi-objective learning-based heuristic called LSEO and compared it to several recent and state-of-the art algorithms from the literature.

The results show the viability of the proposed sustainable DPFSP. First of all, the feasibility of the developed optimization model has been shown by a numerical example as given in Figure 1. The optimal Pareto solutions for solving the case of the company CWP have been shown in Figure 6 to confirm the optimality of our solutions compared to the exact solver using the EC method. The high performance of the proposed LSEO was shown in different criteria (Figures 3.7 and 3.8) to confirm its superiority over other algorithms. The variations of the sustainability objectives are illustrated in Figure 3.9. Finally, the efficiency of the optimization model developed was analyzed by certain sensitivity analyses as indicated in Figures 3.10 to 3.12.

From the results, some recommendations can be suggested. First, this study conceptually shifts the energy-efficient DPFSP to the sustainable DPFSP to simultaneously cover all the economic, environmental and social factors. The use of different production technologies can be defined as an introduction to the reverse production and supply chains with multiple production centers. A high number of Pareto solutions found by algorithms gives production managers this possibility to find an interaction between economic, environmental and social alternatives. Last but not least, setting the parameters of the model such as the company's budget or the social weights, is very important to achieve the environmental and social sustainability for a production system. In the continuation of this work, we will try to obtain data from an actual industrial case study, and develop simple and accessible guidelines to help production managers to implement the concept of triple bottom line for production systems.

In conclusion, although this study is more complex than the majority of existing papers in the area of DPFSP, there are many suggestions to continue this line of research as follows:

- Uncertain factors in the definition of DPFSP may be used. The use of robust and stochastic optimization concepts can be suggested to resolve the uncertainty.
- Adding risk factors based on economic, environmental and social criteria to the DPFSP is rarely considered and can be suggested.
- The application of the proposed algorithm to other combinatorial optimization problems such as home healthcare systems and facility location planning, as well as the development of this method with more learning and local search techniques, are some of the potential continuations of this article.

CHAPTER 4

A DISTRIBUTED PERMUTATION FLOW-SHOP CONSIDERING SUSTAINABILTY CRITERIA AND REAL-TIME SCHEDULING

Amir M. Fathollahi-Fard, Lyne Woodward and Ouassima Akhrif,

Department of Electrical Engineering, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper submitted to the *Journal of Industrial Information Integration*, June 2023

Abstract

Recent developments in production scheduling have focused on redefining task scheduling to address real-time events, such as the random arrival of new or unforeseen tasks, and uncertainties, including variations in task processing times and machine breakdowns. These advancements aim to enhance the adaptability and responsiveness of scheduling systems in dynamic production environments. Additionally, sustainable production, which aims to integrate economic, environmental, and social criteria into production scheduling, has emerged as an active challenge in this field. This paper contributes to both real-time scheduling and sustainable production fields by redefining the sustainable distributed permutation flow-shop scheduling problem. The proposed model prioritizes minimizing the makespan while reducing energy consumption, and the number of lost working days, and simultaneously increasing job opportunities within allowable limits. The study considers machines that can operate under different modes, ranging from manual to automatic. In the proposed distributed permutation flow-shop, real-time scheduling is performed using two strategies: predictive-reactive and proactive-reactive scheduling. Two rescheduling policies, continuous and event-driven, are also considered. To demonstrate the applicability of the optimization model, a numerical case study on auto workpiece production is provided. To address the complexity of the model,

various reformulations and heuristics are defined. Lagrangian relaxation and an efficient Benders decomposition reformulation are proposed to solve the model initially without considering real-time events. Additionally, four problem-specific heuristics are employed to efficiently identify approximate solutions while considering real-time events. A comprehensive analysis and discussion of the results are presented, highlighting the main findings for production managers. Notably, the predictive-reactive scheduling strategy exhibits a slight advantage over the proactive-reactive approach in terms of real-time scheduling analysis for the sustainable distributed permutation flow-shop problem. Furthermore, the event-driven rescheduling policy demonstrates higher efficiency compared to the continuous rescheduling policy. Overall, this study contributes valuable insights into the fields of real-time scheduling and sustainable production. By addressing the challenges of task scheduling in dynamic environments and integrating sustainability criteria, the research provides practical implications for production managers and offers a framework for more efficient and sustainable production systems.

Keywords: Sustainable Production; Distributed Permutation Flow-Shop Scheduling Problem; Uncertainty; Lagrangian relaxation; Benders decomposition; Heuristics;

4.1 Introduction

In today's competitive landscape, manufacturing companies are increasingly striving to establish sustainable production systems that encompass economic, environmental, and social criteria. They seek to integrate these dimensions of sustainability while effectively responding to uncertainties and disruptions in production schedules. Consequently, manufacturing companies are keen on incorporating task assignments on machines within their production scheduling processes, particularly to deal with uncertainty. This enables them to improve sustainability practices and ensure efficient resource allocation in their operations.

State-of-the-art technologies enable various modes of machine operation, ranging from manual to highly automated processes. Thus, for each machine used in task processing, the production manager must select one of these modes with regard to economic, environmental, and social criteria (Varelmann et al., 2022). These modes of operation require different levels of human interaction. For example, a machine can be operated in manual mode which requires significant human involvement, or in various automated modes which reduce the need for human intervention. Moreover, if a new mode of operation is introduced or new operators are hired, the number of lost working days spent training workers on machine operation will also vary depending on the mode of operation selected (Fathollahi-Fard et al., 2021).

This flexibility in operating modes enables production centers to tailor their processes to specific needs and optimize efficiency accordingly. Indeed, unlike a traditional production system where machines are most of time operated in manual mode, an Industry 4.0-based production system can be supported by technologies including advanced automatic modes of operation derived from concepts such as the internet of things and/or cyber-physical systems (Frank et al., 2019). As such, recent advances in industrial informatics and Industry 4.0 are useful in dealing with uncertainty in production scheduling (Dalenogare et al., 2018). In an Industry 4.0-based production system, uncertainties can be managed by the use of real-time scheduling where simulations, optimization, and probabilistic theories are integrated with rescheduling strategies and policies (Ghaleb et al., 2020). Based on these needs of real-time optimization and the benefits of a sustainable production system, this study aims to develop a comprehensive optimization model for a sustainable distributed permutation flow-shop where each factory has several machines to process the tasks in the event of disruptive events. The goal is to find a minimum makespan which is the maximum completion time for all factories.

According to the ISO 14000 standard (Corbett, & Kirsch, 2001), in the manufacturing sector, environmental sustainability is defined as a reduction in carbon emissions and energy consumption. Regarding energy consumption, a machine can have different levels of consumption depending on its status and the selected mode of operation. In this study, the energy consumption of the machines is evaluated according to three different statuses. The first one considers the processing of a task by a machine. The second status refers to the idle time when the machine is powered but waiting for a task to be processed. Finally, the third one is the ultra-low idle status, where the numerical control (NC) device shuts down the servo system of the machine, reducing its power consumption to the lowest level. Compared to its ultra-idle status, a machine that is in its idle status consumes a relatively high amount of energy. It is reported that the energy consumption that is not related to task processing can reach more than 40% of the total energy consumption (Li et al., 2019). Reducing the amount of energy consumed by a machine in an idle status, or minimizing the time spent in an idle status, can effectively improve the energy efficiency of the production process.

The ISO 26000 standard provides a framework for assessing the social performance of manufacturing companies, with a focus on improving the quality of human life (Llach et al., 2015). To this end, social sustainability is a key criterion for evaluating the impact of manufacturing practices on society. In this context, the proposed comprehensive optimization model considers several factors, including the number of job opportunities created and working days lost, as key performance indicators for assessing social sustainability. By incorporating these metrics into the optimization model, the social impact of manufacturing operations can be measured and improved, contributing to a more sustainable and socially responsible approach to manufacturing systems.

Social sustainability has a significant impact on people's lives, particularly in countries where the industrial sector comprises a significant portion of the gross domestic product (GDP). China, for instance, has approximately 80 million employees in the manufacturing sector. As depicted in Figure 4.1, in 2020, the agricultural, industrial, and service sectors accounted for 23.6%, 28.7%, and 47.7% of the workforce, respectively. The industrial sector, which generated nearly 32.6% of China's GDP in 2021, was by far the largest contributor, followed by the wholesale and retail sectors (9.7%) and the financial sector (8.0%). Employment opportunities in the manufacturing industry are influenced by various factors. One of these factors is the mode of operation selected. In manual mode, for example, more workers may be needed compared to automatic modes of production (Fathollahi-Fard et al., 2021). These factors underscore the significance of employment opportunities as a social factor in the context of Industry 4.0, particularly in countries like China, where the industrial sector offers vast employment prospects.

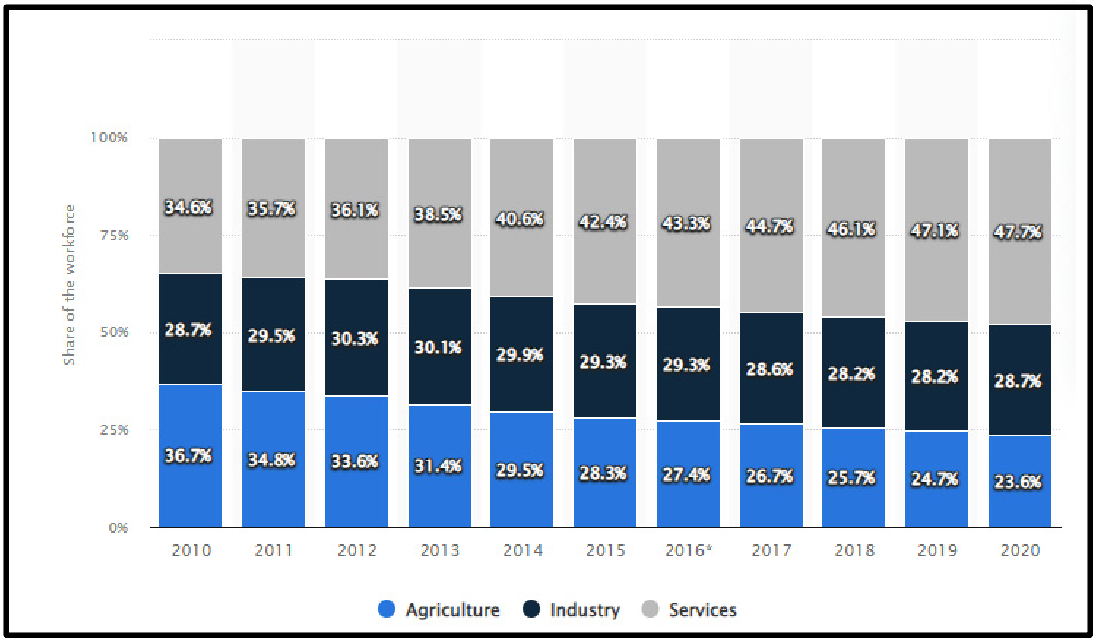


Figure 4.1 Total employment in China from 2010 to 2020 for agriculture, industry (manufacturing sector) and services [4]

To assess social sustainability in the manufacturing industry, the number of workdays lost is an important factor from both economic and social perspectives that must be considered. According to Fathollahi-Fard et al. (2021), the number of working days that operators are unable to work or are restricted from working can be influenced by various reasons. For instance, in 2020, the COVID-19 pandemic severely impacted many workers in China's manufacturing industry due to the high level of risk of contracting the virus[5]. Additionally, a change in the working environment is another reason of lost working day for operators. For example, the introduction of a new automatic operating mode that involves an advanced programming system may require worker training. Operators must be trained to operate the machines in this new mode, while engineers and electricians must become familiar with new programming languages to ensure the maintenance of machines with this specific operating

[4] https://www.statista.com/
[5] http://www.china.org.cn/business/covid-19-economic-impact/node_8018307.html

mode. It is crucial to consider the number of working days lost as an important aspect of social sustainability in the manufacturing industry.

The scheduling of machines and tasks is often challenging due to uncertainties, such as arrivals of tasks, processing times, and machine breakdowns. Fortunately, recent advances in Industry 4.0 technologies and industrial informatics have made it possible to monitor and control these uncertainties in real time (Dalenogare et al., 2018). This allows production systems to react and reschedule tasks without disrupting operations. Real-time scheduling approaches proposed by Manríquez et al. (2022) and Zhuang et al. (2022) enable task reassignment whenever a disruption occurs, such as the sudden arrival of new tasks during the planning horizon. Simulation techniques can be used to map the makespan and the production schedule based on a disruptive event that occurred during the planning horizon (Abreu et al., 2020; Harmonosky & Robohn, 1991). Moreover, the arrival of new tasks caused by the development of new products or even the change of operating mode on a machine make task processing times uncertain. Estimating processing times can be achieved using fuzzy logic or stochastic theory. Machine breakdowns are another source of uncertainty, and probabilistic theories can be used to estimate their occurrence. To manage these uncertainties, this study applies the concept of a real-time scheduling. To this end, the predictive-reactive and proactive-reactive scheduling strategies, as well as the continuous and event-driven rescheduling policies are applied and evaluated. By implementing these approaches, manufacturing companies can better predict, control, and monitor disruptions, while still considering economic, environmental, and social criteria in their sustainable production systems.

As it is widely known, permutation flow-shop scheduling problems are NP-hard optimization problems (Naderi & Ruiz, 2010). Therefore, the literature has devoted significant efforts to the development of different metaheuristic algorithms to tackle these problems. However, the focus of this paper is on reformulation techniques and heuristics for the proposed problem. Our approach has several advantages in comparison with metaheuristics. Firstly, reformulation techniques allow the problem to be solved using existing optimization solvers, such as CPLEX, which can be more efficient and effective than designing and implementing a new metaheuristic algorithm from scratch (Soleimani et al., 2022). This can save valuable time and resources, particularly when dealing with complex scheduling problems (Hamzadayı, 2020). Secondly, reformulation techniques can provide insights into the problem structure and enable the identification of key variables and constraints, leading to a better understanding of the problem and more informed decision-making when designing a solution approach (Fathollahi-Fard et al., 2020). Lastly, reformulation techniques can provide rigorous mathematical guarantees on the quality of the solution obtained (Hamzadayı, 2020). This is particularly important in applications where the solution's quality is critical, such as in manufacturing and logistics. In summary, although there are many solution approaches available for the distributed permutation flow-shop scheduling problem, our use of reformulation techniques, specifically Benders decomposition and Lagrangian relaxation with constructive heuristics, provides a valid and effective approach for solving the proposed problem.

In conclusion, this study defines a comprehensive optimization model to minimize the makespan of a production system while considering notions of sustainability through constraints on energy consumption, the number of jobs created and the number of working days lost. In addition, the proposed model deals with uncertainty in the framework of real-time scheduling. As our contributions to the modeling of this production scheduling system lead to a complex optimization problem, efficient methods are needed to solve it. The main highlights of the paper are summarized hereafter:

- A comprehensive optimization model for the distributed permutation flow-shop taking into account sustainability criteria and real-time scheduling is developed;
- Efficient reformulations based on Lagrangian relaxation and Benders decomposition as well as problem-specific heuristics are introduced to solve the proposed optimization problem.

The rest of this article is summarized as follows. Section 4.2 collects and reviews the relevant papers in the area of production scheduling with respect to uncertainty, sustainability and distributed permutation flow-shops. Section 4.3 defines the proposed problem, the assumptions and the optimization model. Section 4.4 proposes solution methods based on problem-specific heuristics, Lagrangian relaxation, and Benders decomposition methods. Section 4.5 presents computational tests, validation, comparison and sensitivity analyses for different rescheduling policies and strategies. Finally, Section 4.6 concludes our main findings, recommendations, and future research avenues.

4.2 Literature review

The field of production scheduling has been well-studied during the last century and includes many significant contributions (Graves, 1981; Tang et al., 2001; Gahm et al., 2016). In order to highlight the main useful contributions related to the sustainable distributed flow-shop scheduling problem incorporating Industry 4.0 concepts, the literature review is divided into two subsections. We first review the main models defined and used in the context of Industry 4.0 with emphasis on the management of uncertainty in production scheduling. Then, we present studies on the distributed permutation flow-shop scheduling. We finally identify research gaps that prompted this study.

4.2.1 Production scheduling under uncertainty

In the context of Industry 4.0, new advances in the field of production scheduling open the door to smart production systems (Rossit & Tohmé, 2018; Parente et al., 2020). Based on empirical research, Rossit et al., (2019a) defined how production scheduling is influenced by Industry 4.0 concepts. In another interesting survey, Zhang et al., (2019) collected real-time data and evaluated a set of job-shop scheduling models developed in the context of Industry 4.0. Another survey by Dolgui et al., (2019) discussed the application of optimal control in production scheduling, supply chain and Industry 4.0-based systems. The review paper reports the main contributions, applications and recommendations for these systems.

Using Industry 4.0 concepts and the optimization theory applied to a production system, uncertain production scheduling problems considering a disruption event such as the arrival of a new task (Rahmani & Ramezanian, 2016; Shen & Yao, 2015; Gao et al., 2015) or variable processing times (Framinan et al., 2019) have been studied in the literature. Rescheduling refers to a process in which an existing production schedule is updated in response to such disruptive events. It is defined by three basic terms: strategies, policies and methods. A rescheduling strategy describes whether updated production schedules are generated or not, while a rescheduling policy specifies when and how rescheduling is done (Framinan et al., 2019). Finally, different rescheduling methods are used to update schedules. Table 4.1 shows a classification of strategies, policies and methods used in real-time scheduling as established by Ghaleb et al., (2020).

Table 4.1 Real-time scheduling concepts

Strategies	**Policies**			**Methods**
	When-to-reschedule	**How-to-reschedule**		
		Fixed sequencing	**Rescheduling**	
• Completely-reactive scheduling • Predictive-reactive scheduling • Proactive-reactive scheduling	• Continuous rescheduling • Periodic rescheduling • Event-driven rescheduling	Right shift rescheduling	• Partial rescheduling • Complete rescheduling	• Dispatch rules • Optimization algorithms • Simulation-based scheduling • Machine learning-based scheduling

Real-time scheduling methods can be applied as part of any of the three scheduling strategies of Table 4.1 (completely-reactive, predictive-reactive, and proactive-reactive scheduling strategies). Under the completely-reactive scheduling strategy, no firm schedule is known in advance. Therefore, decisions are made locally in real-time. A dispatch rule is used to select a task with the highest priority from a set of tasks as the next to be processed by a machine as soon as it becomes available (Framinan et al., 2019). The main difference between proactive-reactive and predictive-reactive scheduling strategies is that the former develops stochastic schedules while the latter develops deterministic schedules. A stochastic schedule considers real-time events like the random arrival of new tasks, while the deterministic case has no real-time events. However, both cases consider the effects of machine failures using fuzzy, stochastic or probabilistic theory to estimate the duration of these failures. Machine downtime prediction is called preventive maintenance in the context of production scheduling (Gholizadeh et al., 2021).

The predictive-reactive scheduling has two main steps. The first one is to generate an initial production schedule considering a deterministic problem. The second step considers the uncertain disruptive events and then updates the current task schedule before such events occur in order to reduce their impact on production system performance (Ghaleb et al., 2020). Finally, the proactive-reactive scheduling develops stochastic schedules where the deterministic problem (i.e., the scheduling model without a disruption) is updated once a disruptive event has occurred (Ghaleb et al., 2020). The results available in the literature (Ghaleb et al., 2020; Framinan et al., 2019) confirm that predictive-reactive and proactive-reactive scheduling strategies are less time-consuming compared to completely-reactive scheduling. In our real-time scheduling approach for the distributed permutation flow-shop system both predictive-reactive and proactive-reactive scheduling strategies are considered and compared. As given in Table 4.1, three well-known rescheduling policies are used for estimating when to reschedule, including continuous, periodic and event-driven rescheduling policies. Under a periodic rescheduling policy, a schedule is revised (or created) periodically over time, while the event-driven policy reschedules when certain events occur, including the arrival of rush orders and cancellation of orders (Al-Behadili et al., 2020). The literature shows that an event-driven policy results in a shorter makespan than a periodic rescheduling policy (Framinan et al., 2019).

Furthermore, continuous rescheduling is a special case of event-driven rescheduling, since its approach consists in rescheduling the production each time an uncertain event occurs such as the arrival of tasks or the failure of a machine (Liu et al., 2017). A continuous rescheduling policy involves regular and ongoing updates to the schedule based on real-time information. On the other hand, an event-driven rescheduling policy reacts to a subset of all specific triggers or events that necessitate a schedule update. These triggers can be changes in task dependencies, resource availability, unexpected events, or disruptions. In our distributed permutation production scheduling, this study compares the continuous rescheduling policy with the event-driven rescheduling policy.

Regarding how-to-reschedule policies, there are different approaches to rescheduling tasks in response to disruptions. One method is the right shift rescheduling policy, which involves postponing each remaining operation by the time needed to make the schedule feasible (Rahmani & Ramezanian, 2016; Shen & Yao, 2015). Partial rescheduling is another option, which involves rescheduling only the affected operations while leaving the remaining tasks unchanged. Full rescheduling is a third option, which entails recomputing the entire schedule using optimization algorithms. In the case of partial rescheduling, the affected operations are identified and rescheduled while keeping the rest of the schedule intact (Gao et al., 2015). Generally, the solution obtained from a full rescheduling is expected to be better than that resulting from a partial rescheduling because the full rescheduling consists in optimizing the entire schedule by considering all the tasks and their dependencies. On the other hand, partial rescheduling focuses only on rescheduling the affected tasks and may not consider the overall schedule optimization. Hence, this study uses a comprehensive optimization approach to perform full rescheduling.

Real-time scheduling can be achieved using several methods, including dispatch rules, optimization algorithms, simulation-based techniques, and machine learning-based algorithms. For instance, dispatch rules manage manufacturing systems by selecting tasks to be assigned to machines as they become available, without generating a production schedule (Rahmani & Ramezanian, 2016). Dispatch rules always offer a local optimal solution, but an optimization algorithm can find the global optimal solution iteratively (Fathollahi-Fard et al., 2021). However, optimization algorithms and dispatch rules require the definition of an optimization problem, whereas simulation-based and machine learning-based techniques do not rely on such problems.

Among the most relevant papers related to real-time scheduling, one of the earliest works was by Shen & Yao (2015). These authors defined the problem of flexible job-shop scheduling as a multi-objective optimization problem including criteria such as energy-efficiency and task assignment stability. They then solved this problem using an evolutionary algorithm. Gao et al., (2015) solved the same problem using a two-stage artificial bee colony algorithm. In this approach, the first stage generates an initial task schedule while the second, initiated upon the arrival of a new task, performs task rescheduling. Rahmani & Ramezanian (2016) investigated flexible job-shop scheduling allowing new potential task arrivals by defining a multi-objective optimization problem considering tardiness and stability of scheduling operations as objectives. A solution for this problem was obtained using the variable neighborhood search algorithm. As another example, Fu et al., (2018) developed a flow-shop scheduling model to minimize the total makespan and tardiness. Given the context of Industry 4.0, they include the time needed to train workers exposed to new cutting-edge technologies in the model. They managed uncertainty using stochastic parameters describing machine processing time and worker learning curves. To solve this multi-objective problem, they applied a fireworks algorithm and compared the results with that obtained using a non-dominated sorting genetic algorithm, a multi-objective evolutionary algorithm based on decomposition and a multi-start simulated annealing algorithm. Han et al., (2018) developed a blocking lot-streaming flow-shop scheduling model with stochastic processing time for an Industry 4.0 based system. A multi-objective migrating birds' optimization algorithm was developed to solve the proposed model. Framinan et al., (2019) proposed a permutation flow-shop scheduling problem considering variable processing times for machines. They minimized the makespan while evaluating two rescheduling policies, including continuous and periodic rescheduling. Finally, Gholizadeh et al., (2021) proposed a robust optimization problem for the flexible job-shop considering preventive maintenance and solved it using a scenario-based genetic algorithm with new crossover and mutation operators.

Recent studies have typically considered more than one type of disruptive events, such as random task arrivals and machines failures (Ghaleb et al., 2020; Al-Behadili et al., 2020; Liu et al., 2017; Shahrabi et al., 2017). Shahrabi et al., (2017) proposed a job-shop scheduling problem accounting for random task arrivals and machine failures which were handled using an event-driven rescheduling policy. They solved this problem using the variable neighborhood search algorithm improved by the reinforcement learning method. Liu et al., (2017) developed a heuristic solution based on the tabu search algorithm including an event-driven rescheduling policy for solving a mixed-shop scheduling problem which also considered the possibility of arrival of new tasks and machine breakdowns. Al-Behadili et al., (2020) defined a multi-objective permutation flow-shop scheduling problem with multiple disruption events. The main novelty was in the resolution algorithm based on a predictive-reactive approach combining a randomization process and the iterated greedy algorithm. Ghaleb et al., (2020) developed a flexible job-shop scheduling problem with random task arrivals and machines breakdowns with continuous and event-driven rescheduling policies. They minimized the tardiness by a hybrid genetic algorithm having three problem-specific heuristics as decision rules.

4.2.2 Distributed permutation flow-shop models

Considering the makespan as the optimization performance criterion, Naderi & Ruiz (2010) defined and modeled for the first time a distributed permutation flow-shop scheduling problem. As this problem considers more than one factory in the production scheduling, its complexity is higher than more traditional flow-shop scheduling problems that aim to schedule the tasks to be performed in a single factory. In this regard, they proposed two decision rules to heuristically assign tasks to factories and then improve these solutions using variable neighborhood procedures. The same problem was then tackled using different approaches such as a genetic algorithm based on local search strategies (Gao & Chen, 2011), a modified iterated greedy search algorithm (Lin et al., 2013), a scatter search heuristic (Naderi & Ruiz, 2014) or another metaheuristic algorithm inspired by chemical reactions (Bargaoui et al., 2017).

Fernandez-Viagas et al., (2018) also studied the distributed permutation flow-shop scheduling problem, this time considering as the main criterion to be minimized, the total flow-time which is the sum of the completion times of all the factories. Pan et al., (2019) solved the same problem using local search heuristics while Ruiz et al., (2019) proposed a simplified version of an iterated greedy heuristic. Meng et al., (2019) modified the problem by adding the possibility to receive orders from different customers. They developed an evolutionary swarm-based optimization algorithm to solve this new problem.

The research in this field has recently embraced the concept of environmental sustainability. Wang & Wang (2018) introduced an energy-efficient distributed permutation flow-shop scheduling model that aimed to minimize makespan and energy consumption. Fu et al. (2019) addressed this problem using a brainstorm optimization algorithm, while Wang et al. (2020) applied a multi-objective whale optimization algorithm. Lu et al. (2020) incorporated a processing time penalty into their energy-efficient distributed permutation flow-shop problem, considering it a negative social factor. However, social sustainability, as per ISO 26000, encompasses indicators such as job opportunities, lost working days, workplace injuries, and local social development. They employed a multi-objective memetic optimization method and compared the results with those of other well-known algorithms. Fathollahi-Fard et al. (2021) developed a multi-objective sustainable distributed permutation flow-shop scheduling problem, aiming to minimize the makespan, energy consumption, and lost working days while maximizing job opportunities. They proposed a learning-based social engineering optimizer for their deterministic model. Lastly, Yue et al. (2023) addressed energy-efficient scheduling in the printed circuit board manufacturing industry with a bi-objective mathematical model. They proposed a hybrid Pareto spider monkey optimization algorithm and compared its effectiveness with that of other multi-objective evolutionary algorithms.

4.2.3 Research gaps and contributions

In order to clearly identify the gaps in the research that underlies the proposed approach, Table 4.2 and Table 4.3 illustrate the main contributions existing in the literature to date with those provided by this research. The most relevant works were identified as those taking into account more than one sustainability criterion or, with at least one assumption regarding uncertainty. The first criterion used to classify the relevant works in this table is the configuration of the production system under consideration. Then come the sustainability criteria, namely the economic, environmental and social criteria. The economic criterion is itself defined by the makespan, the flow-time and the tardiness. The next category is that related to uncertainty and includes, on one hand, disruptive events, i.e., random arrivals of new tasks, machine failures and variable processing times and, on the other hand, the rescheduling policies used to manage uncertainties (continuous, periodic or event-driven policies). The ability to select more than one production modes on machines is another concept for Industry 4.0 based systems that is considered in this table. Finally, the last criterion considered in Table 4.2 concerns the algorithms and heuristics used to solve the problem. After analyzing the most relevant works in the field and classifying them in Table 4.2 and Table 4.3, the following conclusions can be drawn:

- Only Lu et al., (2020) and Fathollahi-Fard et al., (2021) tried to consider all economic, environmental and social factors simultaneously. However, their model was deterministic and they did not consider energy consumption, job opportunities and lost workdays.
- No paper considered a production system modeled as a distributed permutation flow-shop scheduling problem dealing with multiple uncertainties.
- No paper simultaneously studied all the disruptive events (variable processing time, random task arrivals and machine failures).
- Although one of the features of Industry 4.0 is to use advanced technologies in production systems, with the exception of Fathollahi-Fard et al., (2021), no study has considered the possibility of production mode selection for machines.
- Only a few studies have considered the rescheduling policies (Shahrabi et al., 2017; Liu et al., 2017; Framinan et al., 2019; Ghaleb et al., 2020; Al-Behadili et al., 2020). However, these studies did not take into account environmental and social criteria, two of the three criteria defining sustainability.

Table 4.2 Summary of relevant studies based on sustainability criteria

Reference	Configuration of production systems	Sustainability criteria			Proposed solution algorithm
		Economic	Environmental	Social	
Shen and Yao (2015)	FJSP	✓	✓	-	Modified evolutionary algorithm
Gao et al., (2015)	FJSP	✓	-	-	Two-stage ABC
Rahmani and Ramezanian (2016)	FJSP	✓	-	-	VNS
Shahrabi et al., (2017)	JSP	✓	-	-	VNS with reinforcement learning
Liu et al., (2017)	Mixed-shop	✓	-	-	TS
Wang and Wang (2018)	DPFSP	✓	✓	-	Knowledge-based cooperative algorithm
Fu et al., (2018)	FSP	✓	-	-	Multi-objective fireworks algorithm
Han et al., (2018)	FSP	✓	-	-	Multi-objective migrating birds' optimization
Fu et al., (2019)	DPFSP	✓	✓	-	Brain storm optimization
Framinan et al., (2019)	PFSP	✓	-	-	-
Wang et al., (2018)	DPFSP	✓	✓	-	Multi-objective whale optimization algorithm
Han et al., (2020)	BFS	✓	✓	-	Improved multi-objective evolutionary algorithm
Lu et al., (2020)	DPFSP	✓	✓	✓	Multi-objective memetic algorithm
Al-Behadili et al., (2020)	PFSP	✓	-	-	Iterated greedy algorithm
Ghaleb et al., (2020)	FJSP	✓	-	-	Hybrid GA and Heuristics
Gholizadeh et al., (2021)	FJSP	✓	-	-	Scenario-based GA
Mansouri et al., (2016)	FSP	✓	✓	-	Lower bounds and a heuristic
Mokhtari and Hasani (2017)	FJSP	✓	✓	-	Improved SPEA2
Wu and Sun (2018)	FJSP	✓	✓	-	NSGA-II with heuristics
Wang et al., (2018)	Parallel machine	✓	✓	-	Constructive heuristic
He and Sun (2013)	FJSP	✓	-	-	-
Jing et al., (2021)	DPFSP	✓	-	-	Iterated greedy with local search
Wu and Che (2019)	Parallel machine	✓	✓	-	Hybrid memetic differential evolution algorithm

(Continued)

Reference	Configuration of production systems	Sustainability criteria			Proposed solution algorithm
		Economic	Environmental	Social	
Dai et al., (2019)	FJSP	✓	✓	-	Enhanced GA
Zhang et al., (2019)	Hybrid FSP	✓	✓	-	Three-stage MOEA/D
Tirkolaee et al., (2020)	FSP	✓	✓	-	Self-adaptive artificial fish swarm algorithm
Shukla et al., (2020)	Parallel machine	✓	✓	-	Enhanced multi-objective evolutionary algorithm
Sin et al., (2020)	Parallel machine	✓	✓	-	Hybrid multi-objective GA
Anghinolfi et al., (2020)	Parallel machine	✓	✓	-	Greedy search with local search
Hong et al., (2021)	FSP	✓	✓	-	Improved MOEA/D
Marichelvam and Geetha (2021)	FSP	✓	✓	-	Hybrid memetic with VNS
Jiang et al., (2022)	FJSP	✓	✓	-	Improved ABC
Yue et al. (2023)	DPFSP	✓	✓	-	Improved Pareto-spider monkey optimization
This research	DPFSP	✓	✓	✓	Reformulations, Heuristics and Metaheuristics

To address these research gaps, this study develops a comprehensive optimization model for a sustainable distributed permutation flow-shop accounting for job opportunities and lost working days as well as the possibility of switching the production modes from manual to automatic, each of these modes requiring more or less advanced technology. This paper also uses disruptive factors, including the arrival of new tasks, varying processing times, and machines breakdowns. We also compare continuous and event-driven rescheduling policies which are well used and well established in the production scheduling research field. Another novelty is the development and evaluation of efficient reformulations using Lagrangian relaxation and Benders decomposition techniques to reduce the complexity of our optimization problem. Finally, problem-specific heuristics are proposed to evaluate rescheduling policies.

Table 4.3 Relevant studies based on uncertainty and real-time scheduling criteria

Reference	Uncertainty			Operatin g mode selection	Rescheduling policies		
	Random task arrival	Machine's breakdown	Variable process time		Continuous rescheduling policy	Periodic rescheduling policy	Event-driven rescheduling policy
Shen and Yao (2015)	✓	-	-	-	-	-	-
Gao et al., (2015)	✓	-	-	-	-	-	-
Rahmani and Ramezanian (2016)	✓	-	-	-	-	-	-
Shahrabi et al., (2017)	✓	✓	-	-	-	-	✓
Liu et al., (2017)	✓	✓	-	-	-	-	✓
Fu et al., (2018)	-	-	✓	-	-	-	-
Han et al., (2018)	-	-	✓	-	-	-	-
Framinan et al., (2019)	-	-	✓	-	✓	✓	
Al-Behadili et al., (2020)	✓	✓	-	-	-	-	✓
Ghaleb et al., (2020)	✓	✓	-	-	✓	-	✓
Gholizadeh et al., (2021)	-	✓	✓	-	-	-	-
He and Sun (2013)	-	✓	-	-	✓	-	-
Jing et al., (2021)	-	-	✓	-	-	-	-
Wu and Che (2019)	-	-	✓	-	-	-	-
Tirkolaee et al., (2020)	✓	-	✓	-	-	-	-
Shukla et al., (2020)	-	-	✓	-	-	-	-
Sin et al., (2020)	-	✓	-	-	-	-	-
Marichelvam and Geetha (2021)	-	-	✓	-	-	-	-
This research	✓	✓	✓	✓	✓	-	✓

4.3 Proposed problem

The main objective of the proposed problem is to find an optimal sequence of N tasks to be processed on M different machines that can operate according to P different modes of production and are distributed in F factories. The sequence of these tasks performs O operations. In the following subsections, we formalize the mathematical description of sustainability and uncertainties before presenting the proposed optimization model and the notations used.

4.3.1 Sustainability

The proposed problem includes features related to the economic, environmental and social dimensions defining the concept of sustainability. The problem considers the energy consumption which affect both the economic and environmental criteria, the yield loss which affects the economic criterion, and the number of job opportunities created and working days lost which are linked to the social dimension.

Knowing that machines mainly consume non-renewable energy and generate carbon emissions, it becomes obvious that the amount of energy consumed by the production system is an important concern. To effectively manage the energy consumption, in this study, the amount of energy consumed by the machines of the production system is classified into three levels defined according to the three operating states of these machines. Thus, the amounts of energy required by a machine being respectively in ultra-low idle, idle, or processing states are denoted EC_{mpf}, IEC_{mf} and UEC_{mpf}. All of these energy consumption levels must be below a predefined upper bound ($UBEC$). In addition, each machine can operate in manual or automatic modes. An error rate depending on the chosen operating mode is defined for each machine (RW_{mpf}). An automatic production mode generally creates less wastes than a manual mode.
The social dimension of sustainability is taken into account by defining a number of job opportunities created and working days lost in order to improve the quality of life and the

environment of workers (Fathollahi-Fard et al., 2021). Depending on its mode of production, each machine needs less or more workers to process tasks (JO_{mpf}). A machine operated in manual mode generally requires more workers than in automatic mode. From the point of view of social sustainability, a higher number of workers is favorable. Thus, a lower bound is defined for the number of expected job opportunities created (LBJ). Depending on the mode of operation chosen, the level of knowledge required by operators will vary. For example, a machine operated in advanced automatic mode driven by a programmable logic controller (PLC) (Alphonsus, & Abdullah, 2016) or an automatic position control (APC) system (Shilyaev et al., 2013) may require a period of training for operators. The duration of this training period in days is defined as lost working days (LD_{mpf}). According to economic and social criteria, a reduction of this factor is favorable. Therefore, an upper bound (UBL) defines the allowable number of lost working days.

4.3.2 Uncertainties

As mentioned earlier, this paper models an uncertain production system. Processing time and predicting disruptions such as the arrival of new tasks and machine breakdowns should be estimated to mitigate the effect of these uncertainties on the performance of the production system. First, the variable processing time (PT_{nmpf}) of task *n* on machine *m* operated in mode *p* in factory *f* can be estimated using pessimistic, realistic and optimistic scenarios that neglect the impact of machine breakdown. In this regard, using the fuzzy method of Jiménez, et al., (2007) and considering $PT_{nmpf}^{pes}, PT_{nmpf}^{rea}$ and PT_{nmpf}^{opt} as respectively pessimistic, realistic and optimistic estimations of the processing time, we can define the expected processing time (EPT_{nmpf}) as follows:

$$EPT_{nmpf} = \frac{PT_{nmpf}^{pes}, +2PT_{nmpf}^{rea} + PT_{nmpf}^{opt}}{4}, \forall n \in \mathcal{N}, m \in \mathcal{M}, p \in \mathcal{P}, f \in \mathcal{F}. \quad (4.1)$$

In order to strengthen our estimate, we introduce machine failure and repair rates into the processing time estimate. In the proposed distributed permutation flow-shop system, machines are considered to have two states with respect to their ability to process a task: machine *m* using production mode p is capable to process task n or needs to be repaired. Referring to He, & Sun, (2013) and Mehta, & Uzsoy, (1998), random machine breakdowns follow a probabilistic function with exponential distribution (Ross, 2019). This study assumes that each machine m using production mode p has fixed failure and repair rates, γ_{mp} and δ_{mp} respectively. In this regard, the mean time to failure and the mean time to repair are respectively $\frac{1}{\gamma_{mp}}$ and $\frac{1}{\delta_{mp}}$. Hence, we compute the processing time *(PC_{nmpf})* of operation *(O_{nmpf})* by adding to the expected processing time (EPT_{nmpf}) the production delay caused by machine breakdowns and the time required for repairs using the probabilistic theory of Ghaleb et al., (2020):

$$PC_{nmpf} = EPT_{nmpf} + \left\{\left(TF_{nmpf} + \frac{1}{\delta_{mp}}\right) \times \left(\frac{e^{-\gamma_{mp}EPT_{nmpf}}}{1 - e^{-\gamma_{mp}EPT_{nmpf}}}\right)\right\}, \quad \forall n \in \mathcal{N}, m \in \mathcal{M}, p \in \mathcal{P}, f \in \mathcal{F} \tag{4.2}$$

where TF_{nmpf} is the time of a failure to occur within EPT_{nmpf} and is estimated as follows:

$$TF_{nmpf} = \frac{\frac{1}{\gamma_{mp}}\left(1 - e^{-\gamma_{mp}EPT_{nmpf}}\right) - EPT_{nmpf}e^{-\gamma_{mp}EPT_{nmpf}}}{1 - e^{-\gamma_{mp}EPT_{nmpf}}}, \quad \forall n \in \mathcal{N}, m \in \mathcal{M}, p \in \mathcal{P}, f \in \mathcal{F} \tag{4.3}$$

In addition to process and failure time estimation, the machine state, MS_{mpft}, is set to 1 if the machine is busy on an operation. Otherwise, a maintenance or repair plan can be performed when the machine is not busy with an operation ($MS_{mpft} = 0$). RP_{mpft} defines the time during which a machine is repaired (i.e., the machine is not busy on an operation$MS_{mpft} = 0$) while AV_{mpft} represents the time which is necessary for the machine to process a task (i.e., the machine is busy on an operation $MS_{mpft} = 1$). Last but not least, we can process a task on a machine if it is able to process it (H_{nimpft}).

Thus, the proposed model is designed to handle two types of uncertainties in production scheduling: machine breakdowns and the arrival of new tasks. Machine breakdowns are addressed by using probabilistic theory to estimate the downtime of machines. On the other hand, real-time scheduling is used to handle new task arrivals over a time horizon starting at time *t*, the time when the new task arrives in the production system. In this study, a deterministic schedule refers to a schedule where no real-time event occurs ($t = 0$). However, the model also covers stochastic scheduling, which includes real-time scheduling in the operational planning horizon ($t > 0$) where one or more real-time events take place with random arrivals of new tasks. This allows the model to be flexible in adapting to various types of production environments to ensure optimal scheduling under uncertain conditions.

4.3.3 Decision variables and the objective function

Two main decision variables are defined in the proposed problem:

- The mode of operation selected for each machine (Y_{mpf});
- The assignment of tasks to each machine (X_{nimpft}) defining its sequence;

These two binary variables define the search space of our optimization problem. Moreover, four additional auxiliary decision variables are linked to these main decision variables as follows:

- The expected time at which a task will begin to be processed on a machine (S_{impft}) is related to the sequence of tasks on this machine (defined by variables X_{nimpft})
- The expected number of tasks assigned to each factory (A_{ft}) is defined using the task assignment variable (X_{nimpft});
- The completion time (C_{impft}) of each task depends on the task assignment variable (X_{nimpft}), and the expected start time of a task (S_{impft});
- The expected time for completing tasks in a factory (CT_{ft}) is calculated using task completion times (C_{impft}).

The objective function is to minimize the expected total makespan for all factories ($CMAX_t$), which depends on the expected time for completing tasks in the factories (CT_{ft}). This means that the model prioritizes minimizing the total time required to complete all tasks, which is a key goal of production scheduling (Naderi & Ruiz, 2010). However, social and environmental sustainability factors such as energy consumption, lost working days, and job opportunities are also considered and treated as constraints in the scheduling model. Thus, the model takes them into account when generating schedules, while still prioritizing minimizing makespan. If these factors were included as objective functions, the optimization problem would become overly complex and difficult to solve. Additionally, the model may not be able to generate a schedule that's optimal for all objectives. By treating them as constraints, the optimization problem remains manageable, and the solution is more likely to be feasible from a practical perspective.

4.3.4 Notations and the problem formulation

Before establishing our optimization problem, all indices, parameters and decision variables are briefly defined hereafter:

Indices:

f	Index of factories, $f \in \mathcal{F} = \{1, 2, \dots, F\}$;
m	Index of machines, $m \in \mathcal{M} = \{1, 2, \dots, M\}$;
n	Index of tasks, $n \in \mathcal{N} = \{1,2, \dots, N\}$;
p	Index of operating modes, $p \in \mathcal{P} = \{1,2, \dots, P\}$;
i	Index of task positions in a schedule, $i \in \mathcal{N} = \{1,2, \dots, N\}$;
t	The time at which a real-time event takes place;

Parameters:

B	Maximum budget allowed for the installation of machines and operating modes in the production system, including the salary of workers (in \$);
CO_{mpf}	Cost of operating machine *m* in mode *p* in factory *f* (in \$);
JO_{mpf}	Number of job opportunities created by the use of machine *m* according to mode of operation *p* in factory *f*;
CJ_{mpf}	Salary of workers operating machine m in mode p in factory *f* (in \$) during the planning horizon;
LD_{mpf}	Number of days needed to train the workers to operate machine m in a new operating mode p in factory *f*;
MW	Maximum waste ratio allowed in all factories;

RW_{mpf} Waste ratio of machine *m* when operated in mode *p* in factory *f;*

O_{nmpf} Operation defined as the process of task *n* by machine *m* operating in mode *p* in factory *f;*

PC_{nmpf} Expected processing time of operation O_{nmpf} (in *hours*);

UEC_{mpf} Amount of useful energy consumed by machine *m* operating in mode *p* in factory *f (*in *kWh);*

EC_{mpf} Amount of energy consumed by machine *m* in mode *p* in factory *f* during the total time period it is in the ultra-low idle status *(*in *kWh);*

IEC_{mf} Amount of energy consumed by machine *m* in factory *f* during the total time period it is in the idle status *(*in *kWh)*

$UBEC$ Upper bound of energy consumption *(*in *kWh);*

LBJ Lower bound of the number of job opportunities generated;

UBL Upper bound of the number of lost working days;

MS_{mpft} Status of machine *m* whose operating mode *p* is selected in factory *f* at time *t*; it equals to 1 if the machine is processing a task; otherwise, 0;

AV_{mpft} Time (in *hours*) where a machine *m* whose operating mode *p* is selected in factory *f* at time *t* is necessary to process a task; it is a positive value if the machine is busy on an operation ($MS_{mpt} = 1$); otherwise, 0;

RP_{mpft} Time (in *hours*) needed by machine *m* whose operating mode *p* is selected in factory *f* at time *t* to recover; it is a positive value if the machine fails at time t ($MS_{mpt} = 0$); otherwise, 0;

H_{nimpft} It gets 1 if machine *m* whose operating mode *p* is selected in factory *f* at time *t* is capable to process operation O_{nmpf}; otherwise, 0.

Decision variables:

Y_{mpf} If the production mode *p* is assigned to machine *m* in factory *f*, 1; otherwise, 0;

ST_{impft} Expected starting time (in *hours*) of task processing at position *i* on machine *m* whose operating mode *p* is selected in factory *f* at time *t;*

X_{nimpft} If the task *n* is set at position *i* on machine *m* whose operating mode *p* is selected in factory *f* at time *t*, 1; otherwise, 0;

A_{ft} Expected number of tasks assigned to factory *f* at time *t*. This is an auxiliary variable depending on X_{nimpft};

C_{impft} Expected completion time (in *hours*) of a task at position *i* on machine *m* whose operating mode p is selected in factory *f* at time *t*. This is an auxiliary variable depending on X_{nimpft} and ST_{impft};

CT_{ft} Expected time (in *hours*) for completing tasks in factory *f* at time *t.* This is an auxiliary variable depending on C_{impft};

$CMAX_t$ Expected makespan (in *hours*) for completing tasks in all factories at time *t*. This is an auxiliary variable depending on CT_{ft} for each factory $f \in \mathcal{F}$.

Using these notations, a mixed integer linear programming model addressing the sustainability dimensions and uncertainties with real-time scheduling capabilities is now developed.

$$Z = \min(E(CMAX_t = \max(CT_{ft}))) \tag{4.4}$$

s.t.

$$\sum_{m=1}^{M}\sum_{p=1}^{P}\sum_{f=1}^{F}(Y_{mpf} \times JO_{mpf} \times CJ_{mpf}) + \sum_{m=1}^{M}\sum_{p=1}^{P}\sum_{f=1}^{F}(Y_{mpf} \times CO_{mpf}) \leq B \tag{4.5}$$

$$\sum_{m=1}^{M}\sum_{p=1}^{P}\sum_{f=1}^{F}(Y_{mpf} \times RW_{mpf}) \leq MW \tag{4.6}$$

$$\sum_{i=1}^{N}\sum_{f=1}^{F}X_{nimpft} = 1, \qquad \forall n \in \mathcal{N}, m \in \mathcal{M}, p \in \mathcal{P}, time\ t \tag{4.7}$$

$$\sum_{n=1}^{N}\sum_{f=1}^{F}X_{nimpft} = 1, \qquad \forall i \in \mathcal{N}, m \in \mathcal{M}, p \in \mathcal{P}, time\ t \tag{4.8}$$

$$\sum_{n=1}^{N}\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{p=1}^{P}(X_{nimpft}) = A_{ft}, \qquad \forall f \in \mathcal{F}, time\ t \tag{4.9}$$

$$\sum_{n=1}^{N}\sum_{i=1}^{N} X_{nimpft} < N \times Y_{mpf}, \qquad \forall m \in \mathcal{M}, p \in \mathcal{P}, f \in \mathcal{F}, time\ t \tag{4.10}$$

$$\sum_{p=1}^{P} Y_{mpf} = 1, \qquad \forall m \in \mathcal{M}, f \in \mathcal{F} \tag{4.11}$$

$$X_{nimpft} \leq H_{nimpft} \qquad \forall i, n \in \mathcal{N}, m \in \mathcal{M}, p \in \mathcal{P}, f \in \mathcal{F}, time\ t \tag{4.12}$$

$$ST_{i,m,pft} \geq \sum_{n=1}^{N}(X_{nimpft} \times \{MS_{mpft}AV_{mpft} + \left(1 - MS_{mpft}\right)RP_{mpft}\}), \tag{4.13}$$

$$\forall i \in \mathcal{N}, m \in \mathcal{M}, p \in \mathcal{P}, f \in \mathcal{F}, time\ t$$

$$C_{impft} \geq ST_{i,m-1,pft} + \sum_{n=1}^{N}(X_{nimpft} \times PC_{nmpf}), \tag{4.14}$$

$$\forall i \in \mathcal{N}, m > 1, p \in \mathcal{P}, f \in \mathcal{F}, time\ t$$

$$C_{impft} \geq ST_{i-1,mpft} + \sum_{n=1}^{N}(X_{nimpft} \times PC_{nmpf}), \tag{4.15}$$

$$\forall i > 1, m \in \mathcal{M}, p \in \mathcal{P}, f \in \mathcal{F}, time\ t$$

$$CT_{ft} \geq \sum_{i=1}^{I}\sum_{m=1}^{M}\sum_{p=1}^{P} C_{impft}\,, \qquad \forall f \in \mathcal{F}, time\ t \tag{4.16}$$

$$\sum_{m\in M}\sum_{p\in P}\sum_{f\in F}(Y_{mpf}\times EC_{mpf})+\sum_{m\in M}\sum_{p\in P}\sum_{f\in F}\sum_{n\in N}UEC_{mpf}\times Y_{mpf}\times PC_{nmpf}$$
$$+\sum_{m\in M}\sum_{f\in F}IEC_{mf}\text{ x }\left(\sum_{p\in P}Y_{mpf}\right)\leq UBEC \tag{4.17}$$

$$\sum_{m=1}^{M}\sum_{p=1}^{P}\sum_{f=1}^{F}(Y_{mpf}\times JO_{mpf})\geq LBJ \tag{4.18}$$

$$\sum_{m=1}^{M}\sum_{p=1}^{P}\sum_{f=1}^{F}(Y_{mpf}\times LD_{mpf})\leq UBL \tag{4.19}$$

$$A_{ft}, ST_{impft}, C_{impft}, CT_{ft}, CMAX_t \geq 0 \tag{4.20}$$

$$Y_{mpf}, X_{nimpft}\in\{1,0\} \tag{4.21}$$

The objective function minimizes the makespan, i.e., the maximum completion time in all factories as given in Eq. (4.4) by adjusting the decision variables which are bounded by Eqs. (4.20) and (4.21). The objective function is subject to the set of constraints (4.5) to (4.19), the meaning of which is provided hereafter:

Meaning of constraints:

Constraint (4.5)	Costs of implementing a production mode on a machine in addition to the salary of workers in all factories must not exceed the maximum budget;
Constraint (4.6)	The total ratio of broken products must be lower than a predefined ratio;
Constraints (4.7) to (4.8)	The schedule of tasks must be unique;
Constraints (4.9)	Calculation of the number of tasks assigned to each factory at time *t;*
Constraints (4.10)	Decision variables for the selection of production modes on machines are linked to the decision variables for the assignment of tasks defining the sequence;
Constraints (4.11)	For each machine, a production mode must be selected;
Constraints (4.12)	The assignment of tasks must be performed according to the capability of each machine to process the task;

Constraints (4.13)	The expected starting time of a task must be set once the time of operation or recovery (after a failure) has been completed on the machine;
Constraints (4.14) and (4.15)	The completion time of a task must consider the tasks schedules on machines;
Constraints (4.16)	The expected time for completing tasks in a factory is computed;
Constraints (4.17) to (4.19)	The amount of energy consumed by machines, the number of job opportunities created and the number of working days lost are bounded;

The incorporation of sustainability and uncertainties in the proposed comprehensive optimization model makes it more complex than the traditional distributed permutation flow-shop scheduling model. Indeed, it includes more constraints such as constraints (4.5), (4.6), (4.10) and (4.11) for the selection of operating modes, constraints (4.12) and (4.13) for disruptions, for example, machine breakdowns and arrival of new tasks and, constraints (4.17) to (4.19) for sustainability dimensions. This greater complexity motivates us to develop efficient reformulations and heuristics to effectively solve the proposed optimization problem.

4.4 Proposed solution methods

The best way to deal with a complex NP-hard model is to reformulate it to reduce its complexity (Rahmaniani et al., 2017). Here, two different types of reformulations are developed for this purpose: Lagrangian relaxation and Benders decomposition. Although the proposed reformulation models are efficient for solving small-scale problems, they are still time-consuming when large-scale problems are tackled (Soleimani et al., 2022). More particularly, the reformulations can be used to find a deterministic schedule while they are not able to deal with a real-time schedule (needed to manage a real-time event occurring at $t > 0$). Real-time scheduling necessitates the integration of simulation and optimization. However, traditional exact solvers are not equipped to perform real-time simulations. This is the reason why Benders decomposition and Lagrangian relaxation reformulations are inadequate for handling real-time events in the context of our online optimization problem. To solve large-scale problems and find a real-time stochastic schedule, simple and fast heuristics are introduced. In what follows, we first explain our heuristics (Section 4.4.1) and then present our

Lagrangian relaxation (Section 4.4.2) and Benders decomposition (Section 4.4.3) reformulations.

4.4.1 Heuristics

This study proposes four different problem-specific heuristics for creating a schedule and rescheduling tasks. Although our reformulations can only find a deterministic schedule ($t = 0$), our heuristics are able to deal with uncertainty in real-time by providing a new schedule each time an event occurs (at $t > 0$). The first step in our approach is to process with the selection of operating modes for each machine (Y_{mpf}). Then, we determine the sequence of tasks (through the variable X_{nimpft}) to be processed on the machines before and after the time ($t > 0$) at which the disruptive event occurs. These binary variables build the search space of our optimization problem. Other decision variables which are non-binary ones are computed based on their relations with these two main binary variables from constraints (4.9), (4.10), (4.13), (4.14), (4.15) and (4.16).

To illustrate the proposed approach, consider the following example consisting of two production centers, denoted by F_1 and F_2. Each production center has two machines with different operating modes: M_1 and M_2 are located in F_1, while M_3 and M_4 are located in F_2. The proposed approach assigns operating modes and task sequences to each machine based on a real-time event which occurs at time *t*.

Figure 4.2 depicts a possible solution for this example which is divided into two parts. Figure 4.2(a) illustrates the assignment of operating modes to the machines. In this example, M_1 and M_4 operate in automatic mode, while M_2 and M_3 operate in manual mode. Figure 4.2(b) shows the sequence of 10 tasks assigned to the machines and factories. As can be seen, we can map the assigned tasks based on the time at which the real-time event occurs. For example, the tasks 10, 4 and 6 are assigned to the first factory (F_1) after the time *t* at which the disruptive event occurs. It is worth noting that the first part of the solution definition (Figure 4.2(a)) is common to all heuristics proposed in the study, whereas the second part (Figure 4.2(b)) is obtained by different decision rules, which vary depending on the selected algorithm.

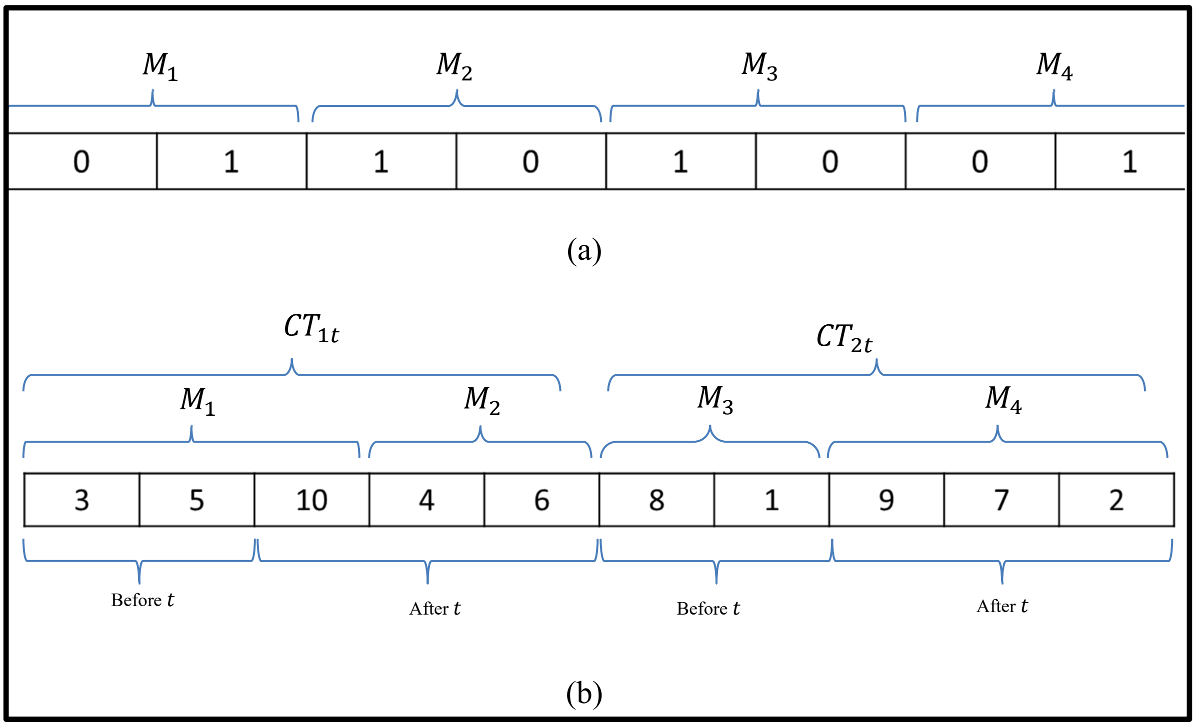


Figure 4.2 Solution definition, i.e., (a) assignment of operating modes, (b) sequence of tasks assigned to each machine

The first step in any of the proposed heuristics is to select the operating mode on every machine in each factory. For this purpose, the following steps must be performed:

- **Step 0**: For each operating mode, each machine, and each factory, compute the average processing time of jobs ($\sum_{n\in N} PC_{nmpf} / N$).
- **Step 1**: For each machine in each factory, select the operating mode (Y_{mpf}) leading the lowest average processing time.
- **Step 2**: Check if constraints (4.5), (4.6), (4.17), (4.18) and (4.19) are satisfied. If they are not satisfied, do the repairing process in the following steps. Otherwise, go to **Step 8**.

- **Step 3**: If constraint (4.5) is not satisfied, identify the machine having the maximum implementation cost. Change the selection of the operating mode for this machine and see if the implementation cost is reduced. Repeat this step for the machine with the highest implementation cost among those not yet tested until this constraint is satisfied.
- **Step 4**: If constraint (4.6) is not satisfied, identify the machine having the highest waste rate. Change the selection of the operating mode for this machine and see if the waste rate is reduced. If so, keep this selection. If not, switch to the mode of operation previously selected for this machine. Repeat this step for the machine with the highest waste rate among those not yet tested until this constraint is satisfied.
- **Step 5**: If constraint (4.17) is not satisfied, identify the machine with the highest total energy consumption $(UEC_{mpf} + EC_{mpf} + IEC_{mf})$. Change the selection of the operating mode for this machine and see if the total energy consumption is reduced. Repeat this step until this constraint is satisfied.
- **Step 6**: If constraint (4.18) is not feasible, identify the machine requiring the smallest number of employed workers (JO_{mpf}). Change the selection of the operating mode for this machine and see if the number of employed workers is increased. If so, keep this selection. If not, switch to the mode of production previously selected for this machine. Repeat this step until this constraint is satisfied.
- **Step 7**: If constraint (4.19) is not feasible, identify the machine leading to the largest number of lost working days. Change the selection of the operating mode for this machine and see if the number of lost working days is reduced. If so, keep this selection. If not, switch to the mode of production previously selected for this machine. Repeat this step until this constraint is satisfied.
- **Step 8**: If there is no infeasibility in constraints (4.5), (4.6), (4.17), (4.18) and (4.19), send the decision variables for the operating mode selection (Y_{mpf}) to the scheduling and rescheduling phases of our heuristics. Otherwise, go back to **Step 2**.

The scheduling phase of our heuristics is based on NR1 and NR2 decision rules proposed by Naderi & Ruiz (2010) for the assignment of tasks in a factory. From the original definition of these decision rules, NR1 and NR2 are respectively fixed as follows:

- Assignment of task *n* to the factory that had the smallest makespan before including this task.
- Assignment of task *n* to the factory that would have the smallest makespan once this task is included.

In both rules above, the first machine to become available in that factory is considered to process the task. However, these decision rules cannot be applied to our proposed sustainable distributed permutation flow-shop scheduling problem due to the different production modes and randomness in machine breakdowns and task rescheduling.

In our heuristics, there are two main phases, i.e., before and after time *t,* when a disruption occurs. NR1 and NR2 are suitable before time *t.* To define the schedule of tasks after time *t*, we have created two decision rules based on the failure recovery time (RP_{mpft}) which affects the expected start time in the constraint set (4.13) at time *t*. These are AF1 and AF2 which are respectively defined below:

- In each factory, assign task *n* to the machine that has the lowest failure recovery time and calculate the makespan without including task *n*.
- In each factory, assign task *n* to the machine that has the lowest failure recovery time and calculate the makespan including task *n*.

In conclusion, by performing Steps 0 to 8 and using decision rules NR1 and NR2 before time *t* and AF1 and AF2 after time *t*, four heuristics called H1, H2, H3, and H4 are respectively defined as follows:

- **H1**: Determine (Y_{mpf}) from Steps 0 to 8. Before time *t*, apply NR1 to establish the task assignment sequence (X_{nimpft}) and compute the makespan ($CMAX_t$). After time *t*, apply AF1 to determine the task assignment sequence and compute the makespan.

- **H2**: Determine (Y_{mpf}) from Steps 0 to 8. Before time *t*, apply NR2 to establish the task assignment sequence (X_{nimpft}) and compute the makespan ($CMAX_t$). After time *t*, apply AF1 to determine the task assignment sequence and compute the makespan.
- **H3**: Determine (Y_{mpf}) from Steps 0 to 8. Before time *t*, apply NR1 to establish the task assignment sequence (X_{nimpft}) and compute the makespan ($CMAX_t$). After time *t*, apply AF2 to determine the task assignment sequence and compute the makespan.
- **H4**: Determine (Y_{mpf}) from Steps 0 to 8. Before time *t*, apply NR2 to establish the task assignment sequence (X_{nimpft}) and compute the makespan ($CMAX_t$). After time *t*, apply AF2 to determine the task assignment sequence and compute the makespan.

4.4.2 Lagrangian relaxation reformulation

The Lagrangian relaxation reformulation aims to relax a set of hard inequality constraints from the original model by adding them into the objective function as soft constraints using Lagrange multipliers (Fisher, 1981). In the case of a minimization problem, the relaxed model provides a lower bound solution which is infeasible for the original problem. Two main criteria are defined to evaluate the quality of the solution obtained from a Lagrangian relaxation reformulation, namely the CPU time to find this lower bound solution and its optimality gap (OG) from the exact solution of the optimization problem (Gmys et al., 2020). Based on these criteria, the best reformulation to find a lower bound is selected. Then, an iterative algorithm is used to update the Lagrangian relaxation multipliers in order to improve this lower bound (Anghinolfi et al., 2020). In this regard, this algorithm aims to minimize the deviation between the updated lower bound and a fixed upper bound which is found by the best solution from NR1 and NR2 which are able of finding a feasible solution satisfying all the constraints without disruptive events. This iterative process ends each time a feasible lower bound is found or the maximum number of iterations is reached.

The most important step in finding an efficient Lagrangian relaxation reformulation is the selection of the set of constraints to be relaxed. If constraints (4.14) and (4.15) are considered to be relaxed with respect to the original model, the Lagrangian relaxation reformulation is as follows:

$$LB = \min\Bigg(CMAX_t + \sum_{i=1}^{N}\sum_{m=2}^{M}\sum_{p=1}^{P}\sum_{f=1}^{F} \pi_{impft}\left(ST_{i,m-1,pft} + \sum_{n=1}^{N} (X_{nimpft} \times PC_{nmpf}) - C_{impft} \right) + \sum_{i=2}^{N}\sum_{m=1}^{M}\sum_{p=1}^{P}\sum_{f=1}^{F} \varphi_{impft}\left(ST_{i-1,m,pft} + \sum_{n=1}^{N} (X_{nimpft} \times PC_{nmpf}) - C_{impft} \right)\Bigg) \tag{4.22}$$

s.t. Constraints (4.5) to (4.13) and (4.16) to (4.21)

where π_{impft} and φ_{impft} are the Lagrange multipliers. We call this reformulation LG1. In addition to this reformulation, other reformulations are defined in Table 4.4. These reformulations are established by relaxing hard constraints that have a significant impact on the computation time compared to the original model. To compare the performance of these Lagrangian relaxation reformulations, four sustainable distributed permutation flow-shop scheduling test problems from Fathollahi-Fard et al., (2021) are used. The CPLEX[6] solver is used to solve all reformulations as well as the original problem. It should be noted that when we apply the CPLEX solver, only deterministic schedule is available. Therefore, this comparison is performed assuming that no real-time event occurs. In order to have a fair comparison, the initial values of Lagrange multipliers were fixed at 1 in all the reformulations.

[6] https://www.ibm.com/analytics/cplex-optimizer

Each time, CPU time is calculated on a laptop with Intel(R) Core (TM) i7-10850H CPU @ 2.70GHz 2.71 GHz.

Table 4.4 Lagrangian relaxation reformulations

Reformulation model	Relaxed constraints
LG1	Constraint sets (4.14) and (4.15)
LG2	Constraint set (4.14)
LG3	Constraint set (4.15)
LG4	Constraints (4.5) and (4.6)
LG5	Constraint set (4.17)
LG6	Constraints (4.18) and (19)
LG7	Constraints (4.17), (4.18) and (4.19)
LG8	Constraints (4.5), (4.6) and (4.17) to (4.19)

The results are shown in Table 4.5, where the CPU time is provided in seconds. The OG is expressed as the relative deviation of the lower bound solution found using the reformulation from the exact solution of the original problem. Thus, a lower value of OG means a better solution. Figure 4.3 draws the difference in performance of the 8 reformulations as a function of the CPU time and OG criteria. Based on the CPU time criterion, unsurprisingly, all reformulations are easier to solve than the original problem. Among the reformulations, LG1 is the fastest and LG5 is the slowest one. Moreover, based on the optimality gap, LG1 shows the best performance while LG8 is the weakest model. In conclusion, the LG1 reformulation is the most efficient Lagrangian reformulation to solve our optimization problem.

We propose four problem-specific heuristics to solve the proposed model. These heuristics provide the upper bound (UB) to our Lagrangian reformulation. Hence, the Lagrangian reformulation LG1 comes in four versions, each based on one of the heuristics. The Lagrange multipliers given in Eq. (4.22) are increasingly updated as follows:

$$\pi_{impft}^{it+1} = max\left(\pi_{impft}^{it} + f^{it} \times \frac{(UB - LB^{it})}{(UB - LB^{it+1})^2} \times \left[(ST_{i,m-1,pft} + \sum_{n=1}^{N}(X_{nimpft} \times PC_{nmpf}) - C_{impft})\right], 0\right) \tag{4.23}$$

$$\varphi_{impft}^{it+1} = max\left(\varphi_{impft}^{it} + f^{it} \times \frac{(UB - LB^{it})}{(UB - LB^{it+1})^2} \times \left[(ST_{i-1,m,pft} + \sum_{n=1}^{N}(X_{nimpft} \times PC_{nmpf}) - C_{impft})\right], 0\right) \tag{4.24}$$

where π_{impft}^{it} and φ_{impft}^{it} are the Lagrange multipliers at iteration *it*. UB is the fixed upper bound identified as the best solution among the solutions found using heuristics H1 to H4. Moreover, LB^{it} is the lower bound solution found at iteration *it* while f^{it} is a scalar number between zero and two generated randomly (Tordecilla et al., 2023). Interested readers who would like to have more details about the approach used to update the Lagrange multipliers, are referred to Tautenhain, et al., (2021) as a good example.

Table 4.5 Comparison of Lagrangian relaxation reformulations in their ability to find a deterministic schedule

Methods	Tests	T1	T2	T3	T4
	Size of tests (*F***M***P***N*)	2*2*2*4	2*2*2*8	2*4*2*20	3*4*3*30
Exact solution	CPU time (s)	10.98	14.75	34.72	65.74
LG1	CPU time (s)	4.65	5.34	12.75	18.39
	OG	0.14	0.23	0.23	0.18
LG2	CPU time (s)	7.95	8.74	23.33	26.11
	OG	0.14	0.29	0.34	0.32
LG3	CPU time (s)	7.03	10.79	21.93	39.84
	OG	0.14	0.32	0.29	0.37
LG4	CPU time(s)	9.33	12.68	26.04	55.22
	OG	0.43	0.56	0.49	0.52
LG5	CPU time (s)	10.16	13.57	30.64	57.19
	OG	0.38	0.42	0.56	0.44
LG6	CPU time (s)	8.89	11.94	28.12	53.24
	OG	0.52	0.75	0.68	0.71
LG7	CPU time (s)	7.56	10.76	25.38	47.96
	OG	0.69	0.65	0.51	0.79
LG8	CPU time (s)	6.54	7.91	19.32	25.74
	OG	0.82	0.86	0.94	0.78

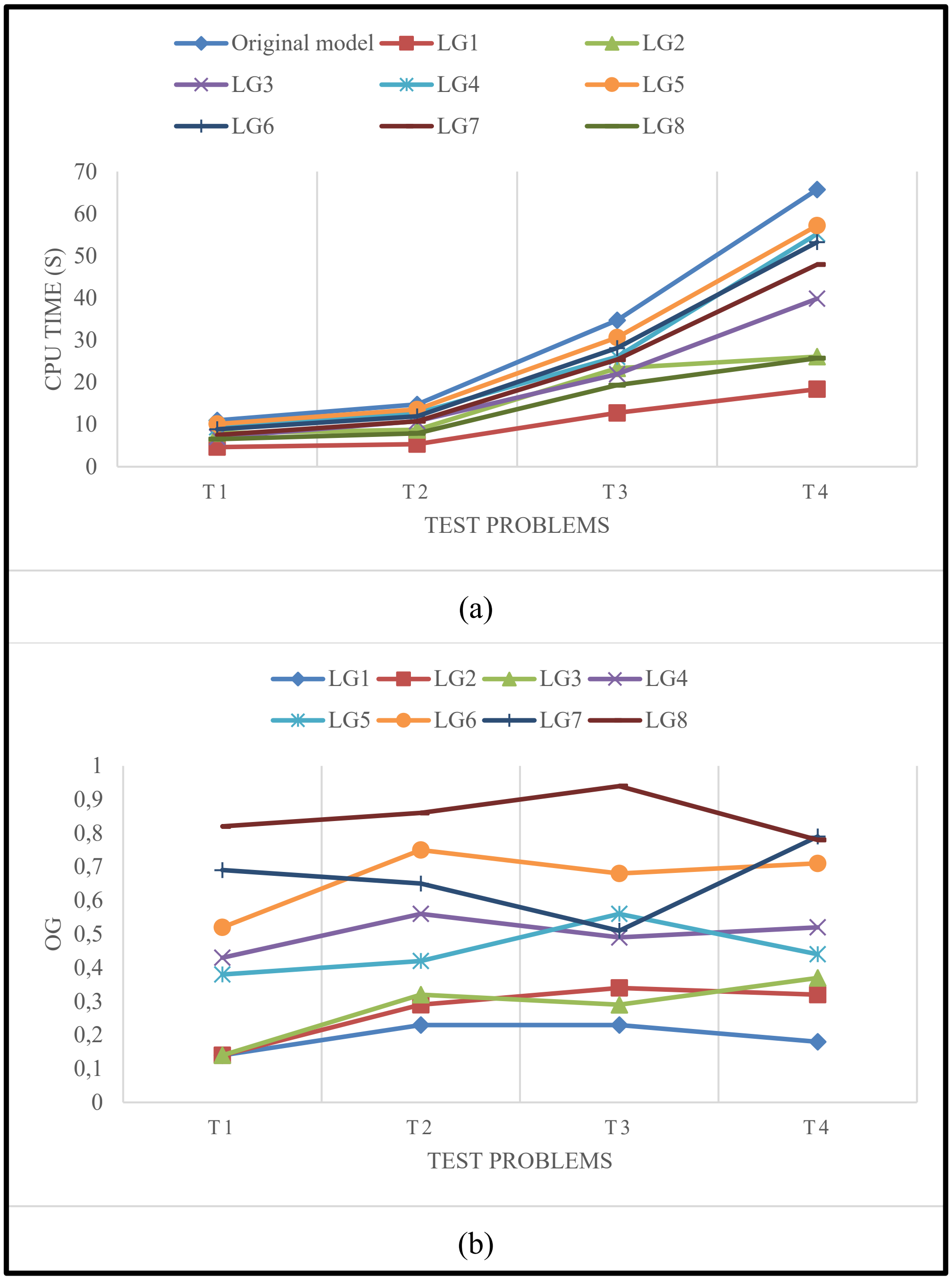


Figure 4.3 Comparison of Lagrangian relaxation reformulations based on the criteria of (a) CPU time and (b) OG

4.4.3 Benders decomposition reformulation

This study also proposes a Benders decomposition (BD) reformulation (Benders, 1962) to solve the original model using a mixed integer programming approach. Indeed, BD reformulation can be applied to an optimization model if both continuous and integer variables exist. This technique is rarely used in the field of production scheduling (Duarte, et al., 2020) due to the many different auxiliary decision variables and constraints that are hardly separable. If an optimization model has fewer decision variables and constraints, BD implementation is easier. Recently, Hamzadayı (2020) proposed an efficient BD reformulation for the traditional distributed permutation flow-shop scheduling problem. However, due to the complexity of the proposed model, this BD reformulation is not applicable in this study. Thus, we propose a new BD reformulation well-adapted to our complex optimization model.

The original optimization model needs to be split into two separate models, namely the master problem (MP) and the primary subproblem (PS). Most notably, the PS model will be solved using a linear programming approach. In this regard, binary variables must be fixed. Consequently, in the PS model, all variables are non-negative continuous variables. Based on a feasible solution for X_{nimpft}, the PS model is formulated as follows:

$$PS = \min(CMAX_t = \max(CT_{ft})) \tag{4.25}$$

$$s.t.$$

$$\begin{gathered} Constraints\ \ (4.9), (4.13), (4.14), (4.15), (4.16), (4.20) \\ A_{ft}, ST_{impft}, C_{impft}, CT_{ft}, CMAX_t\ \geq 0 \end{gathered} \tag{4.26}$$

If X_{nimpft} is feasible, the PS model is also feasible and, its dual is also feasible. For each constraint set of (4.9), (4.13), (4.14), (4.15), (4.16) and (4.20), a continuous decision variable is defined for the dual of PS (DPS). Based on this reformulation, an upper bound for the DPS formulation, is defined as follows:

$$DPS \geq \left(\sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{p=1}^{P} \sum_{f=1}^{F} \beta_{impft} \left(\sum_{n=1}^{N} (X_{nimpft} \times \{MS_{mpft} AV_{mpft} + (1 - MS_{mpft}) RP_{mpft}\}) \right) + \sum_{i=1}^{N} \sum_{m=2}^{M} \sum_{p=1}^{P} \sum_{f=1}^{F} \varepsilon_{impft} \left(\sum_{n=1}^{N} (X_{nimpft} \times PC_{nmpf}) \right) + \sum_{i=2}^{N} \sum_{m=1}^{M} \sum_{p=1}^{P} \sum_{f=1}^{F} \sigma_{impft} \left(\sum_{n=1}^{N} (X_{nimpft} \times PC_{nmpf}) \right) \right) \quad (4.27)$$

An upper bound for the DPS is a lower bound for the PS model (Eq. (4.27) is then a lower bound for the PS model). Thus, Eq. (4.27) is inserted into the MP model to provide an optimality cut for the BD reformulation as follows:

$$MP = \min(CMAX_t) \quad (4.28)$$

$$s.t.$$

$$Constraints\ (4.5), (4.6), (4.7), (4.8), (4.10), (4.11), (4.12), (4.17), (4.18), (4.19), (4.21)$$

$$CMAX_t \geq \left(\sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{p=1}^{P} \beta_{impft} \left(\sum_{n=1}^{N} (X_{nimpft} \times \{MS_{mpft} AV_{mpft} + (1 - MS_{mpft}) RP_{mpft}\}) \right) + \sum_{i=1}^{N} \sum_{m=2}^{M} \sum_{p=1}^{P} \varepsilon_{impft} \left(\sum_{n=1}^{N} (X_{nimpft} \times PC_{nmpf}) \right) + \sum_{i=2}^{N} \sum_{m=1}^{M} \sum_{p=1}^{P} \sigma_{impft} \left(\sum_{n=1}^{N} (X_{nimpft} \times PC_{nmpf}) \right) \right), \forall f \in \mathcal{F} \quad (4.29)$$

$$CMAX_t \geq 0 \quad (4.30)$$

where β_{impft}, ε_{impft} and σ_{impft} are fixed values obtained from the solution of the dual of the PS problem. After solving the MP problem, again the variable X_{nimpft} is sent to the PS model to update the optimality cut. In this respect, iteratively, the MP model calls the PS model and updates the optimality cut and the binary variables. It goes without saying that the MP model provides an efficient BD reformulation for the original model given in Section 4.3. In fact, each subproblem of this reformulation has fewer constraints than the LG1 reformulation presented in Section 4.4.2, which makes the BD reformulation more efficient for solving large-scale test datasets. As far as we know, as our comprehensive optimization model has not been studied before, the proposed BD reformulation is contributed for the first time.

4.5 Computational results

In addition to the small-scale size test problems presented in Section 4.4.2, we also consider in this section, medium and large-scale tests as benchmarked by Fathollahi-Fard et al., (2021). Table 4.6 shows the size of each test problem as well as the initial values of the Lagrange multipliers and the maximum allowable number of iterations used to run both the Lagrangian relaxation and the Benders decomposition reformulations. The range of values for each parameter benchmarked from Fathollahi-Fard et al., (2021) and Ghaleb et al., (2020) is shown in Table 4.7. It should be noted that the expected processing time is evaluated using Eqs. (4.1) to (4.3).

In particular, our analyses are divided into three sections. First, our heuristics and reformulations are analyzed according to the exact solution of the original model obtained from the CPLEX solver in the deterministic condition. After selecting the best reformulation model, more in-depth sensitivity analyzes are performed to evaluate the proposed comprehensive optimization model through a case study. Then, our heuristics are analyzed in their ability to perform real-time scheduling. Finally, different scheduling strategies and rescheduling policies are analyzed in case of a disruptive event. It should be noted that all codes for our reformulations and heuristics are computed on a laptop with an Intel(R) Core (TM) i7-10850H CPU @ 2.70GHz 2.71 GHz.

Table 4.6 Size of test problems

Complexity level	Number of test studies	Size of the test				Initial value of Lagrange multipliers	Number of iterations for running the reformulations
		Number of factories (F)	Number of machines (M)	Number of production modes (P)	Number of tasks (N)		
Small	T1	2	2	2	4	1	10
	T2	2	2	2	8	1	10
	T3	2	4	2	20	1	10
	T4	3	4	3	30	1	10
Medium	T5	3	6	2	30	10	30
	T6	3	6	3	40	10	30
	T7	4	8	4	30	10	30
	T8	4	8	5	40	10	30
Large	T9	6	12	4	80	10	50
	T10	6	12	5	100	10	50
	T11	8	16	6	80	10	50
	T12	10	16	6	100	10	50

Table 4.7 Values of the parameters

Parameter	Range
PT_{nmpf}^{opt}	*randi*([2, 4],N, M, P, F)
PT_{nmpf}^{rea}	*randi*([4, 6],N, M, P, F)
PT_{nmpf}^{pes}	*randi*([6, 8],N, M, P, F)
CO_{mpf}	*randi*([8, 20],M, P, F)*10^4
JO_{mpf}	*randi*([2, 9],M, P, F)
CJ_{mpf}	*randi*([8, 20],M, P, F)
LD_{mpf}	*randi*([8, 30],M, P, F)
LBJ	*round* (*sum*($JO_{mpf}/3$))
UBL	*round* (*sum*($LD_{mpf} * (\frac{2}{3})$))
RW_{mpf}	*rand*(M, P, F)*0.1
IEC_{mf}	(*randi*([8, 12],M, P, F)+*rand*())*10^5
UEC_{mpf}	(*randi*([2, 7],M, P, F)+*rand*())*10^5
EC_{mpf}	(*randi*([20, 40],M, P, F)+*rand*())*10^5
$UBEC$	*round* (*sum*($(IEC_{mpf} + UEC_{mpf} + EC_{mpf}) * (\frac{2}{3})$))
B	*randi*([*round*(*sum*($JO_{mtf}.* CJ_{mtf} + CO_{mtf}$)/2), *round*(*sum*($JO_{mtf}.* CJ_{mtf} + CO_{mtf}$))])
MS_{mpft}	*round(rand*(M, P)*0.8)
H_{nimpft}	*round(rand*(N, I, M, P, F)*0.9)
RP_{mpft} and AV_{mpft}	**if** $MS_{mpt} == 0$ $RP_{mpft} = normrnd\ (sum(H_{nimpft} * PC_{nmpf}), 2 * sum(H_{nimpft} * PC_{nmpf}))$**else** $AV_{mpft} = exp\left(3 * sum\left(H_{nimpft} * PC_{nmpf}\right)\right)$ **end**
γ_{mp}	$\frac{1}{7 * sum(H_{nimpft} * EPT_{nmpf})}$
δ_{mp}	$\frac{1}{\frac{3}{2} * sum(H_{nimpft} * EPT_{nmpf})}$
MW	**if** *sum*(RW_{mpf})>1 *randi*([*round*(*sum*(RW_{mpf})/2), *round*(*sum*(RW_{mpf}))]) **else** *rand*()+(*sum*(RW_{mpf})/2)**end**

**randi*, *rand*, *normrnd, round, sum, exp* are taken from MATLAB function definitions.

4.5.1 Comparison of reformulations and heuristics

First of all, we compare our reformulations with each other to highlight their performance. For the small-scale tests, the original model as well as the two reformulations were solved using the CPLEX software within a reasonable time. As mentioned earlier, we can apply the LG1 and BD reformulations to find a deterministic schedule (there is no real-time events in this case). However, in this comparison, heuristics are able to generate a stochastic schedule in real-time whenever an event occurs at time t within a user-defined time range. Here, $0 < t \leq 5$, i.e. $t =$ *randi*([1, 5])**rand*. It is obvious that these random events make the makespan to be longer than the one found by the deterministic schedules obtained using our reformulations. However, the idea of comparing these random schedules with our reformulations is to study the quality of the deterministic schedules. Moreover, the solutions resulting from the LG1 and BD reformulations suggest a lower bound while the solutions resulting from the heuristics are upper bounds for the proposed problem. As reported in Table 4.8, for small-scale tests the reformulations are compared with each other and with the exact solution of the original problem. We also reported the results of heuristics where a disruptive event, the arrival of a new task, occurs. It should be noted that for each test problem a same disruptive event is simulated to perform an unbiased comparison as the problem size increases. Figure 4.4 shows a comparison for these solutions based on the CPU time and the best makespan identified.

The results obtained show that the two reformulations are faster to solve than the original problem. Moreover, the BD reformulation is solved in a shorter time than the LG1 reformulation, as can be seen in Figure 4.4(a) and Table 4.7. It should be noted that the CPU time required to solve the LG1 reformulation is longer than the CPU time shown in Table 4.5. Here, the iterative algorithm using Eqs. (4.23) and (4.24) is applied whereas the results presented in Table 4.5 where obtained using a fixed upper bound. However, the use of Eqs. (4.23) and (4.24) has no impact on the lower bound found, i.e. the lower bounds of LG1 are the same as those shown in Table 4.5.

Additionally, the BD reformulation produces a stronger lower bound for our model since its solutions are closer to the exact solution, i.e., the optimality gap is lower (as shown in Figure 4.4(b)). Although our proposed heuristics can provide an upper bound for the original model, there is little differentiation among them in this regard. Further analysis is required, as presented in Table 4.9. Overall, our experiments demonstrate that the solution found by the BD reformulation is significantly closer to the exact solution than the one found by the LG1 reformulation. Moreover, BD reformulation solution can be found in a shorter CPU time than the LG1 solution.

BD reformulation is the best method in this comparison because its solution is closer to the optimal solution and it is found in a shorter time than when we solve the original problem. For large-scale tests, it is not possible to identify exact solutions. Consequently, BD reformulation is rather selected as a reference to evaluate and compare the results of our heuristics. As shown in Table 4.8, our heuristics are faster than the proposed BD reformulation. There is no clear difference between the CPU time of our 4 heuristics. Optimal solutions can be expected to be found by the BD (i.e., the deterministic schedules) while our heuristics can be applied after a real-time event occurred. Based on the quality of the solutions of our heuristics evaluated by the optimality gap with the solutions obtained by the proposed BD reformulation, a graph of the means and errors calculated according to a confidence level of 95% and the Student's t distribution is illustrated in Figure 4.5. The data shown on the vertical axis of this graph are obtained by normalizing the OG values of our heuristics with regards to the basic schedules found by the BD reformulation. It should be noted that although the OG values of Table 4 are based on the solution of the original model, in Table 4.9, however these OG values are based on the solution of the BD reformulation which is a lower bound for the original problem. The main conclusions of these computations are that H1 is faster than other heuristics. However, based on Figure 4.5, H4 is very robust and efficient compared to other heuristics. In this comparison, H2 is also better than H3. Finally, H1 turns out to be the worst of all algorithms in this comparison.

Table 4.8 Comparison of reformulations with the original model

Test problem	Original model		LG1 ($t = 0$)		Heuristics ($0 < t \leq 5$)				BD ($t = 0$)	
	Optimal makespan (h)	CPU time (s)	Lower bound (h)	CPU time (s)	Upper bound (h) for H1	Upper bound (h) for H2	Upper bound (h) for H3	Upper bound (h) for H4	Lower bound (h)	CPU time (s)
T1	58.55	10.98	50.31	7.67	72.31	58.55	66.7	62.56	56.347	3.93
T2	121.62	14.75	93.63	8.81	162.38	145.76	156.82	132.4	104.86	4.51
T3	638.43	34.72	491.5	21.03	682.19	675.32	638.43	645.56	550.48	10.78
T4	1009.6	65.74	827.87	30.34	1039.17	1037.86	1042.42	1014.4	927.21	15.55

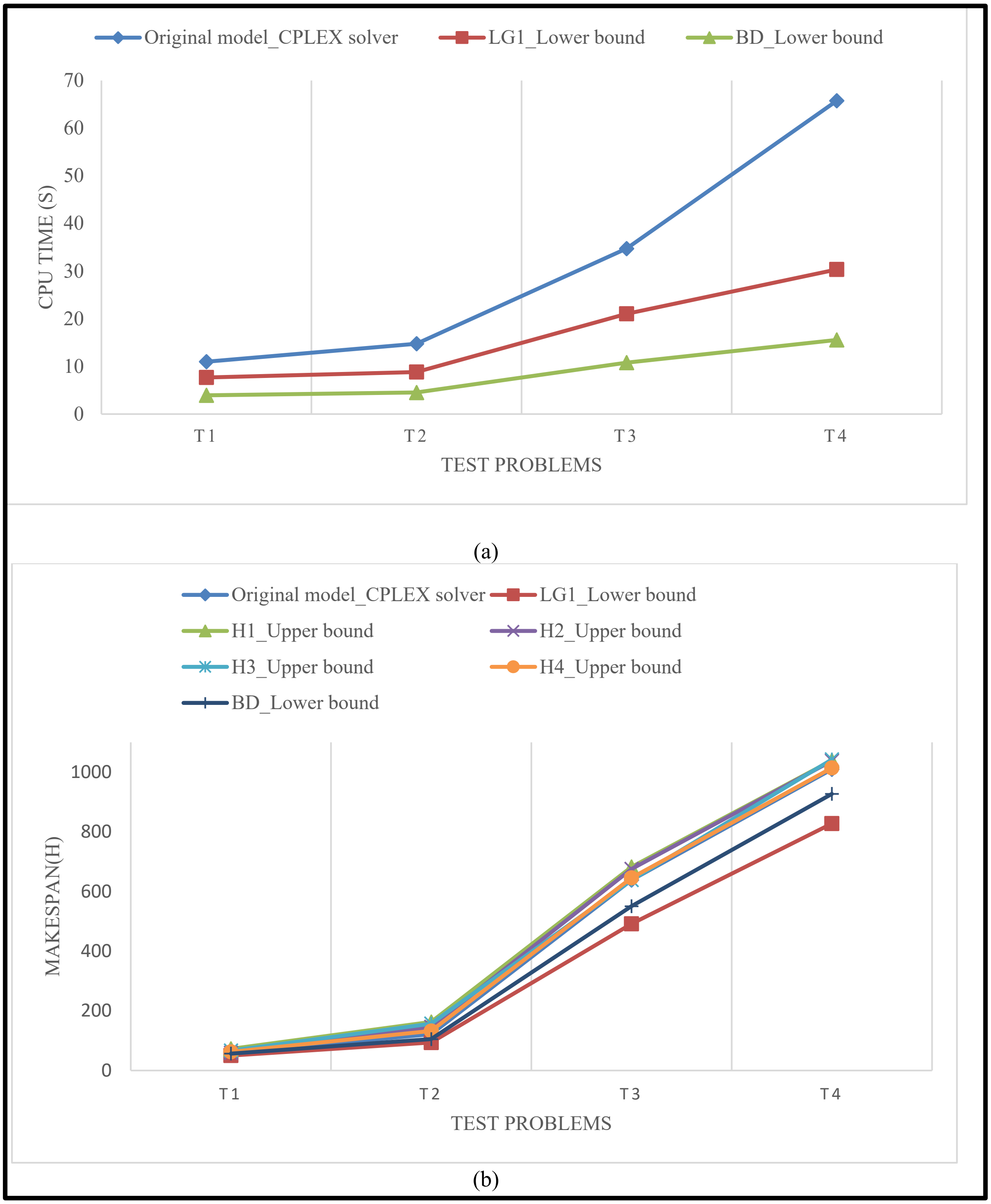


Figure 4.4 Comparison of reformulations for small-scale tests, i.e., T1 to T4 based on CPU time (a) and optimal solutions of the different approaches (b)

Table 4.9 Comparison of heuristics with BD reformulation

Methods		T5	T6	T7	T8	T9	T10	T11	T12
H1	Makespan (h)	196.27	228.54	235.82	319.65	863.11	1093.6	1153.69	1485.95
	CPU time (s)	0.29	0.76	1.93	2.87	10.87	23.76	31.82	41.73
	OG	0.3494	0.1557	0.2861	0.2071	0.1042	0.113	0.1186	0.1357
H2	Makespan (h)	158.35	226.87	198.288	305.75	874.32	1061.65	1136.23	1595.32
	CPU time (s)	0.52	0.98	1.87	3.16	9.64	25.43	36.23	44.62
	OG	0.0886	0.1475	0.0814	0.1546	0.1186	0.0804	0.1016	0.2193
H3	Makespan (h)	167.27	220.55	221.34	285.84	844.28	1142.72	1185.54	1567.54
	CPU time (s)	0.58	1.15	1.52	3.25	12.76	27.97	37.43	48.54
	OG	0.1596	0.1197	0.2034	0.0756	0.0894	0.1633	0.1492	0.1932
H4	Makespan (h)	157.75	213.88	202.81	297.54	856.75	1075.54	1169.5	1412.86
	CPU time (s)	0.67	1.71	2.24	4.89	14.44	28.18	35.89	46.35
	OG	0.084	0.0818	0.1075	0.1264	0.0949	0.0563	0.1394	0.0794
BD	Makespan (h)	145.44	197.75	183.36	264.8	781.6	982.56	1031.36	1308.32
	CPU time (s)	30.99	79.28	103.86	224.85	667.97	6391.69	9060.74	14169.86

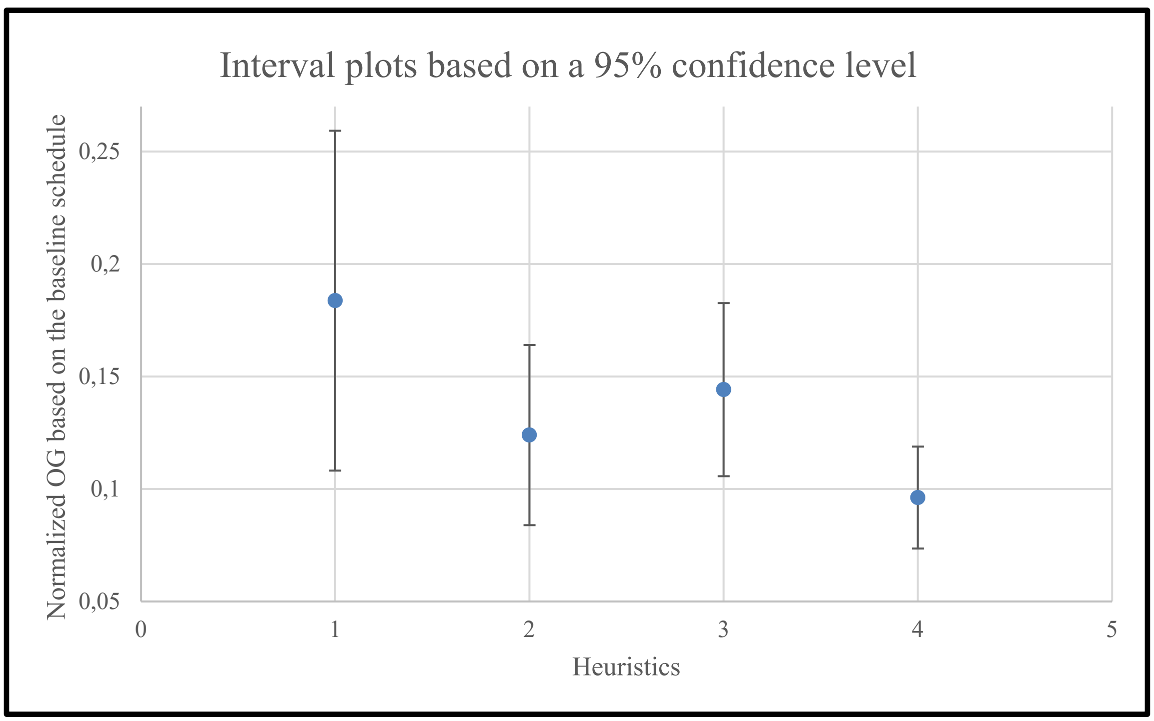


Figure 4.5 Means plot with 95% confidence level for the assessment of heuristics based on the base schedules found by the BD reformulation.

4.5.2 Application of the proposed approach to a case study

Based on the data provided by Wuhan Huazhong Numerical Control Co, a case study was defined to verify the effectiveness of the proposed optimization model. The case study is the production of flanges (Figure 4.6) used in automobile construction. This production consumes a non-negligible amount of energy resulting from the multiple levels of energy consumed by the machines according to three statuses i.e., ultra-idle, idle or processing.

The production of a flange requires ten tasks of turning, milling, drilling, tapping and grinding. These tasks are carried out using five different CNC machines located in two separate factories. Each of the five CNC machines can be operated under 3 different modes of production (Ullah et al., 2021): a manual production mode (MAN) and two automatic production modes based on two advanced operating systems (PLC or APC). In MAN mode, the CNC machine behaves like a conventional or standard machine. In this regard, the operator of a CNC machine is able to press buttons, turn handwheels, and activate switches to operate the process of tasks while improving its functional performance. However, in PLC or APC modes, all of these activities are performed automatically. In order to obtain more information on these CNC machine modes readers can refer to the Wuhan Huazhong Numerical Control Co.[7] as well as studies by Alphonsus & Abdullah (2016) and Shilyaev et al., (2013) which describe PLC and APC systems respectively.

The processing times ($PT_{nmpf} = (PT_{nmpf}^{pes}, PT_{nmpf}^{rea}, PT_{nmpf}^{opt})$) required by the machines to process these ten tasks according to their three different modes of production are shown in Table 4.10. Note that not every task can be processed on every machine. Thus, some rows of this table are empty. Other parameters such as energy consumption levels ($IEC_{mpf}, UEC_{mpf}, EC_{mpf}$) as well as economic and social factors (CO_{mpf} , CJ_{mpf}, JO_{mpf}, LD_{mpf}, RW_{mpf}) are given in Table 4.11. The case study is solved in the context where there is no

[7] https://huazhongcnc.en.made-in-china.com/

disruptive event (t = 0). Thus, only a deterministic schedule is provided where the BD reformulation is used to obtain an optimal makespan of 488.19 minutes in a computational time of 8.45 seconds.

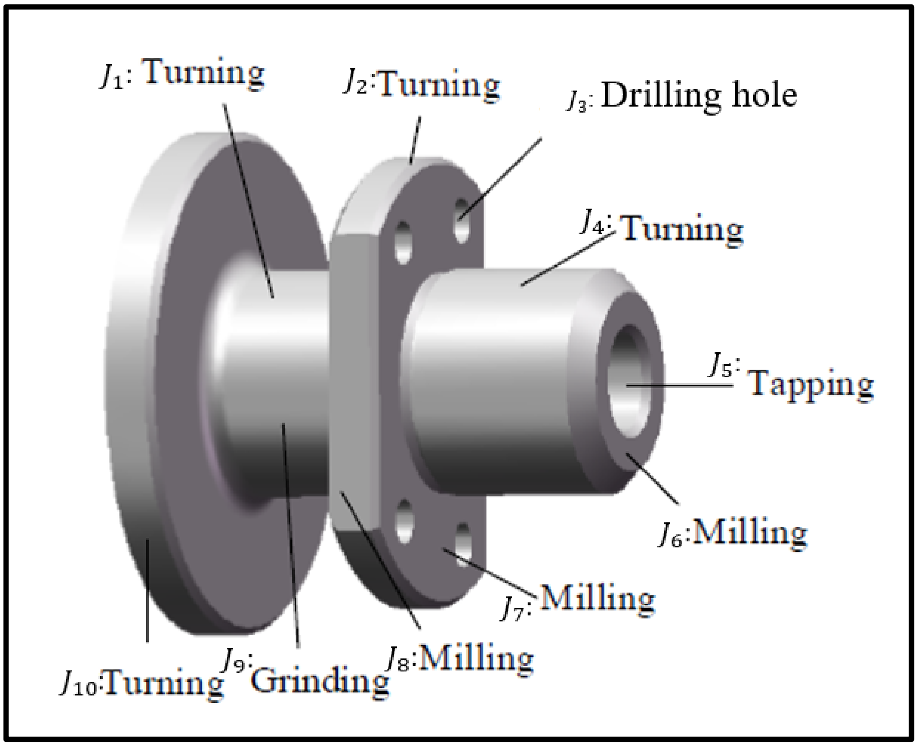


Figure 4.6 The product and its processing tasks in our case study

Table 4.10 Processing times of tasks (PT_{nmpf}^{pes} , PT_{nmpf}^{rea} , PT_{nmpf}^{opt} in minutes)

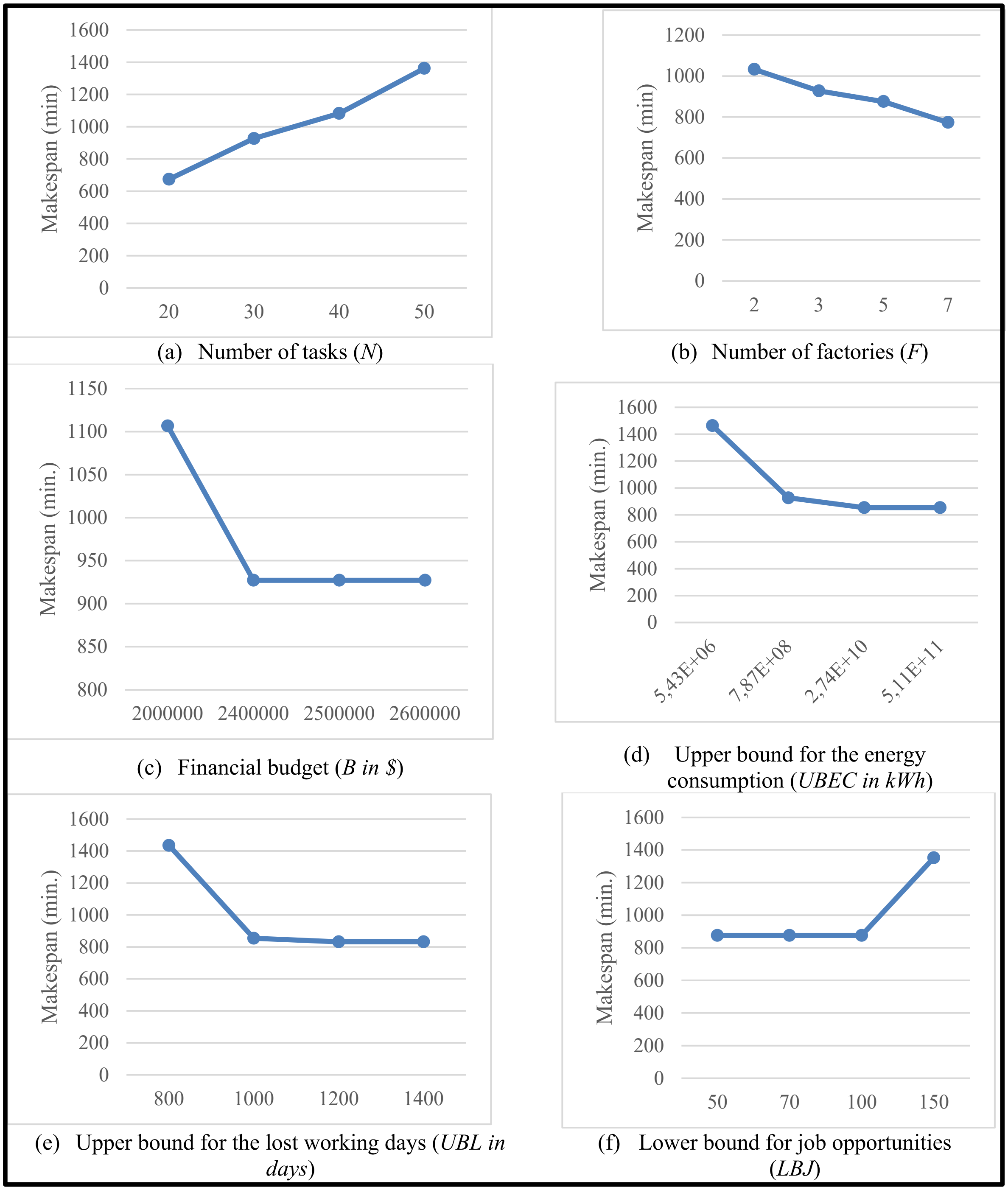
Machine	Production mode	Tasks									
		J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
CNC 1 turning	PLC	(5.6, 5.5, 5.4)	(4.9, 4.8, 4.7)	-	(5.2, 5.1, 5)	-	-	-	-	-	(5.6, 5.5, 5.4)
	APC	(5.6, 5.45, 5.35)	(5, 4.9, 4.8)	-	(5.3, 5.2, 5.1)	-	-	-	-	-	(5.65, 5.5, 5.35)
	MAN	(6, 5.9, 5.7)	(6, 5.8, 5.7)	-	(6.8, 6.5, 6.4)	-	-	-	-	-	(7, 6.7, 6.5)
CNC 2 milling	PLC	-	-	-	-	-	(3.6, 3.5, 3.4)	(3.4, 3.3, 3.2)	(3.5, 3.4, 3.3)	-	-
	APC	-	-	-	-	-	(3.7, 3.6, 3.5)	(3.5, 3.4, 3.3)	(3.6, 3.5, 3.4)	-	-
	MAN	-	-	-	-	-	(6, 5.5, 5)	(6, 5.7, 5.5)	(5.5, 5.3, 5)	-	-
CNC 3 drilling	PLC	-	-	(4.7, 4.6, 4.4)	-	-	-	-	-	-	-
	APC	-	-	(4.8, 4.7, 4.6)	-	-	-	-	-	-	-
	MAN	-	-	(7, 6, 5.5)	-	-	-	-	-	-	-
CNC 4 tapping	PLC	-	-	-	-	(3.9, 3.8, 3.7)	-	-	-	-	-
	APC	-	-	-	-	(4, 3.9, 3.8)	-	-	-	-	-
	MAN	-	-	-	-	(6, 5, 4)	-	-	-	-	-
CNC 5 grinding	PLC	-	-	-	-	-	-	-	-	(4.2, 4.1, 4)	-
	APC	-	-	-	-	-	-	-	-	(4.3, 4.2, 4.1)	-
	MAN	-	-	-	-	-	-	-	-	(6.4, 5.8, 5)	-

Table 4.11 Parameters of our numerical example

Machine	Production mode	IEC_{mpf} (kWh)	UEC_{mpf} (kWh)	EC_{mpf} (kWh)	CO_{mpf} ($)	CJ_{mpf} ($)	JO_{mpf} (Person)	LD_{mpf} (Days)	RW_{mpf}
CNC 1-turning	PLC	0.5	4.1	2.9	32.4×10^3	2	3	7	0.04
	APC	0.45	4.15	3	34.2×10^3	2	3	7	0.03
	MAN	0.52	4.3	3.2	20.4×10^3	1	8	2	0.14
CNC 2-milling	PLC	0.3	3.8	3.1	41.5×10^3	3	2	7	0.02
	APC	0.35	3.75	3.15	42.1×10^3	3	2	7	0.02
	MAN	0.57	4.5	5.2	28.5×10^3	1	6	2	0.12
CNC 3-drilling	PLC	0.2	2.6	1.8	31.2×10^3	3	4	7	0.02
	APC	0.3	2.75	1.9	32.4×10^3	3	4	7	0.01
	MAN	0.4	3.5	2.4	16.3×10^3	2	8	2	0.17
CNC 4-tapping	PLC	0.5	3.1	1.9	23.3×10^3	4	5	7	0.01
	APC	0.45	3.2	2	22.5×10^3	4	5	7	0.03
	MAN	0.75	4.5	3.2	11.7×10^3	2	6	2	0.15
CNC 5-grinding	PLC	0.3	2.6	1.3	32.1×10^3	4	4	10	0.02
	APC	0.35	2.65	1.4	31.7×10^3	4	4	10	0.03
	MAN	0.8	3.76	2.3	18.5×10^3	1	8	3	0.15

In addition to highlight the effect of some key parameters including the number of tasks (N) and factories (F), the budget (B), the upper limits of energy consumption ($UBEC$) and the number of working days lost (UBL) as well as the lower limit of the number of job opportunities created (LBJ) on the results, a sensitivity analysis is performed using different values for the listed parameters. The nominal value of each of these parameters is increased uniformly according to four additional cases (e.g., the nominal value of 10 tasks is increased to 20, 30, 40 then 50 tasks) and for each case, the makespan is evaluated. Figure 4.7 shows the variation of the makespan (in minutes) resulting from the variation of these parameters.

As shown in Figure 4.7(a), an increase in the number of tasks significantly increases the makespan while an increase in the number of factories reduces the makespan (Figure 4.7(b)) as the number of tasks assigned to each factory is reduced. Figure 4.7(c) shows that an increase in the upper bound of the budget can reduce the makespan. However, this improvement is limited. Figure 4.7(d) reveals that increasing the allowable limit of total energy consumption can reduce the makespan. However, as with the upper bound of the budget, this improvement is limited. Figure 4.7(e) confirms that the influence of the upper bound of lost working days on the makespan is generally the same as that of the maximum allowable energy consumption. Finally, as shown in Figure 4.7(f), an increase in the lower bound of the number of job opportunities created does not lead to a reduction of the makespan. However, even if it increases the makespan, its main objective is more of a social nature by offering more job opportunities.



(a) Number of tasks (*N*)

(b) Number of factories (*F*)

(c) Financial budget (*B in $*)

(d) Upper bound for the energy consumption (*UBEC in kWh*)

(e) Upper bound for the lost working days (*UBL in days*)

(f) Lower bound for job opportunities (*LBJ*)

Figure 4.7 Sensitivity analyses on key parameters

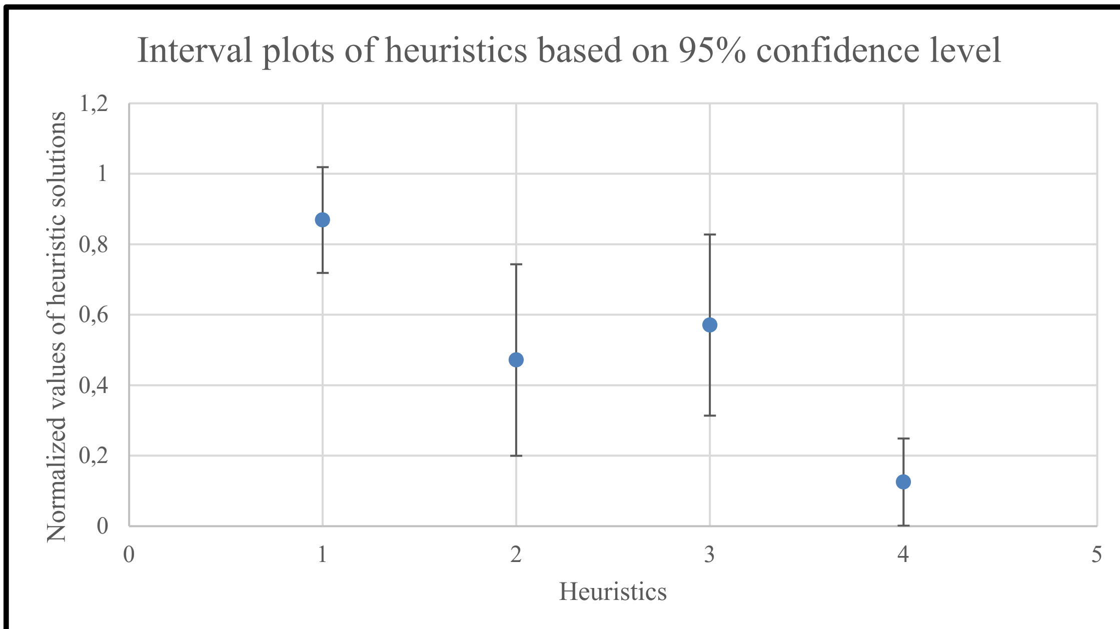
4.5.3 Comparison of scheduling strategies and rescheduling policies

As mentioned previously, the proposed real-time scheduling concept studies the arrival of new tasks at a time *t* while the failure of machines is estimated by probabilistic theories. Predictive-reactive and proactive-reactive scheduling strategies are used to perform this real-time scheduling. Their performances are evaluated and compared according to the makespan. These strategies differ in the application of the decision rules AF1 and AF2 in our heuristics. If a disruptive event is expected at time *t*, the predictive-reactive scheduling strategy applies AF1 and AF2 at time *t-(PCmin)* where *PCmin* corresponds to the minimum processing time among all unassigned tasks. The proactive-reactive scheduling strategy applies the decision rules AF1 and AF2 at time *t*. According to the results presented in Table 4.12, the average of makespan values of all tests obtained using the predictive-reactive strategy is much shorter than those resulting from the application of the proactive-reactive strategy. In order to compare the performance of the 4 heuristics, the makespan values of each test were normalized and the normalized mean values are shown in Figure 4.8 with a 95% confidence level using Student's t-distribution. For the predictive-reactive scheduling strategy (Figure 4.8(a)), the H4 heuristic was found to be the best optimization algorithm. Moreover, H2 is better than H3 while H1 is the worst algorithm of this strategy. For the proactive-reactive scheduling strategy (Figure 4.8(b)), we obtained the same result with H4 and H1 being respectively the best and the worst heuristic in this context.

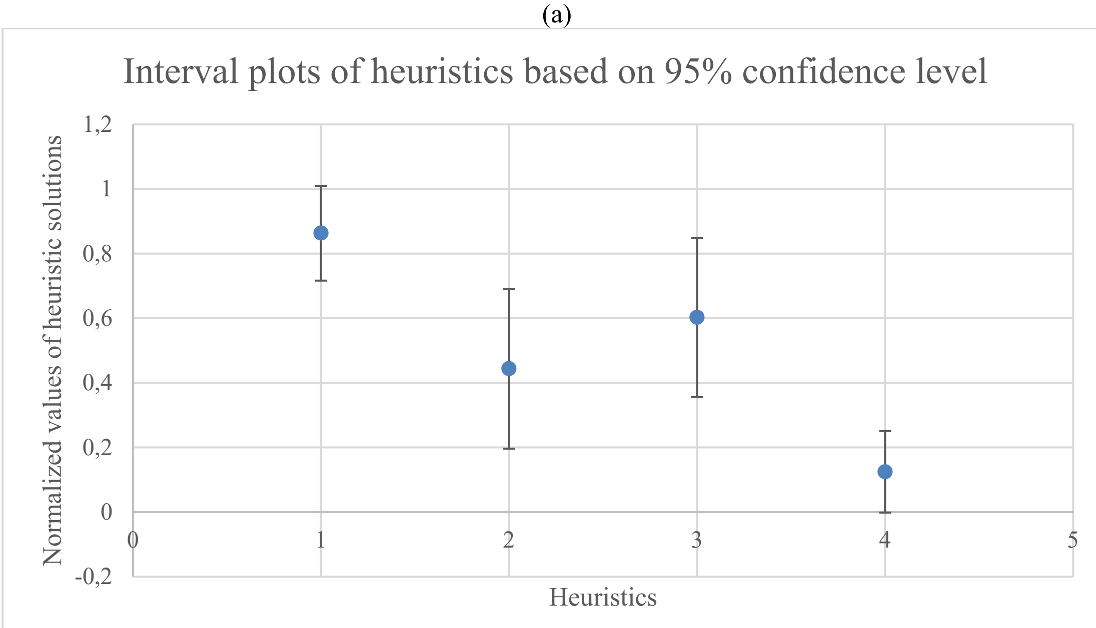
Another important concept is the cost of rescheduling referring to the time lost due to rescheduling. This cost is computed by the deviation of the makespan of the stochastic schedule from the makespan of the deterministic schedule. Table 4.13 reports the comparison of the continuous rescheduling policy with the event-driven rescheduling policy. In each test problem case, we assume that a number of tasks corresponding 25% of the total number of tasks arrive as new tasks at time $0 < t \leq CMAX_t$. with $CMAX_t$ being the makespan at time zero (i.e., the makespan resulting from the deterministic schedule) found by the BD reformulation for each test problem. After generating 0.25 of the total number of tasks in each test problem as real-time events and applying heuristics, the rescheduling cost for each rescheduling policy is computed as the difference in hours between the makespan obtained using the rescheduling policy and the makespan of the deterministic schedule. As the results in Table 4.13 and Figure 4.9 show, continuous rescheduling has a higher cost in terms of time than the event-driven rescheduling policy. Moreover, the behavior of the four heuristics for each of the policies is similar with respect to the solution found.

Table 4.12 Comparison of the makespan (*h*) of the scheduling strategies

Test problem	Predictive-reactive scheduling				Proactive-reactive scheduling			
	H1	H2	H3	H4	H1	H2	H3	H4
T1	80.87	66.07	77.34	65.13	95.27	76.25	92.16	72.22
T2	173.29	156.54	160.15	135.64	187.54	164.19	172.62	145.39
T3	690.36	684.77	647.47	652.06	704.21	693.33	662.34	657.23
T4	1052.13	1050.48	1051.14	1024.66	1067.35	1060.32	1066.86	1033.88
T5	220.88	164.15	176.28	166.92	228.66	170.25	191.65	180.33
T6	242.75	233.09	235.71	223.81	252.58	245.85	247.51	236.84
T7	248.6	216.45	240.59	214.82	258.82	227.38	248.34	225.05
T8	333.94	324.67	303.3	310.58	346.07	331.51	316.53	322.19
T9	879.18	879.49	857.23	871.12	887.95	890.22	866.44	881.97
T10	1120.89	1068.46	1163.17	1080.45	1129.64	1082.63	1170.03	1096.38
T11	1175.88	1155.64	1200.31	1171.67	1187.48	1169.33	1209.22	1182.15
T12	1502.47	1603.35	1577.28	1425.15	1517.96	1610.84	1592.14	1438.57
Average	643.436	633.596	640.83	**611.834**	655.294	643.583	652.986	**622.683**



(a)



(b)

Figure 4.8 Normalized average values of the makespan obtained according to the scheduling strategies, i.e., (a) predictive-reactive scheduling and (b) proactive-reactive scheduling

Table 4.13 Comparison of rescheduling policies based on the cost of rescheduling (*h*)

Tests	Event-driven rescheduling				Continuous rescheduling			
	H1	H2	H3	H4	H1	H2	H3	H4
T1	2.43	1.98	2.32	1.95	7.99	6.62	7.822	5.941
T2	5.2	4.703	4.812	4.075	15.88	14.15	15.16	13.06
T3	20.74	20.57	19.45	19.59	62.48	61.87	59.28	59.03
T4	31.61	31.56	31.58	30.78	95.63	95.26	95.71	93.06
T5	6.637	4.93	5.296	5.01	20.52	15.25	16.29	15.95
T6	7.29	7.003	7.082	6.72	22.08	21.48	21.54	21.03
T7	7.46	6.504	7.22	6.45	22.54	20.21	22.45	19.66
T8	10.03	9.755	9.11	9.33	30.76	30.05	28.25	28.7
T9	26.41	26.42	25.75	26.17	79.67	79.344	77.919	79.2
T10	33.68	32.105	34.95	32.46	101.56	97.074	105.56	97.57
T11	35.33	34.72	36.06	35.2	106.054	104.95	108.796	105.69
T12	45.14	48.17	47.39	42.82	135.894	145.5	142.237	129.35
Average	19.33	19.03	19.25	**18.384**	58.424	57.65	58.416	**55.697**

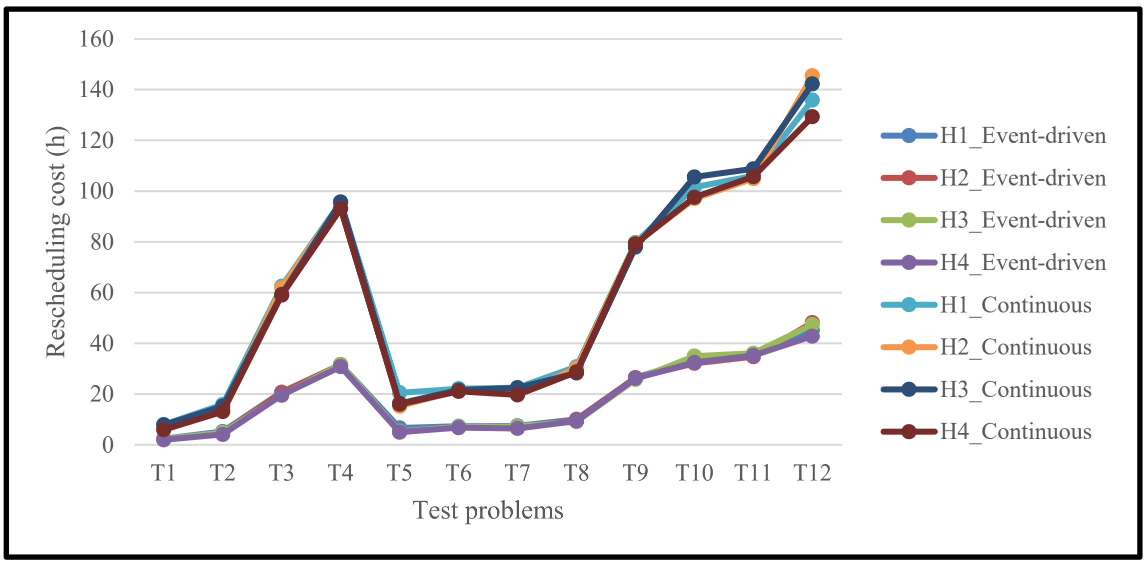


Figure 4.9 Rescheduling cost (in *hours*) of the heuristics according to continuous and event-driven rescheduling policies

In conclusion, the predictive-reactive scheduling strategy is more efficient than the proactive-reactive strategy in performing real-time scheduling for our proposed sustainable distributed permutation flow-shop system. Regarding the how-to-reschedule process, the event-driven rescheduling policy is more efficient than the continuous rescheduling policy. Finally, in all the analyses, our heuristic H4 in majority of analyses, shows the best performance among the other proposed alternatives.

4.6 Discussions, and managerial insights

Recent developments in production scheduling have emphasized the need to redefine task scheduling to accommodate real-time events and uncertainties, such as new task arrivals and machine breakdowns. This shift aims to enhance adaptability and responsiveness in dynamic production environments. Furthermore, the integration of sustainability criteria, encompassing economic, environmental, and social dimensions, has become a pressing challenge in the field. This study contributes by redefining the sustainable DPFSP, aligning it with sustainability goals.

The proposed optimization model prioritizes minimizing the makespan, reducing energy consumption, minimizing lost working days, and increasing job opportunities, all within predefined limits. Real-time scheduling is achieved through predictive-reactive and proactive-reactive strategies, with two rescheduling policies: continuous and event-driven. These approaches represent valuable tools for production managers seeking to balance complex scheduling objectives while responding to dynamic events.

Comparing the performance of reformulations and heuristics highlights their efficiency in finding scheduling solutions. The reformulations, particularly the BD reformulation, prove to be faster and generate solutions closer to the optimal ones. This suggests that incorporating reformulations can significantly reduce computational time and provide strong lower bounds for the DPFSP. The proposed heuristics, while offering upper bounds due to their stochastic nature, demonstrate varying degrees of efficiency. Notably, heuristic H4 stands out as a robust and efficient algorithm.

The sensitivity analysis reveals the impact of key parameters on scheduling outcomes. The number of tasks, factories, budget, energy consumption limits, lost working days, and job opportunities all influence the makespan. For instance, more tasks increase the makespan, while additional factories decrease it. Budget and energy consumption limits can also affect the makespan but with diminishing returns. Job opportunities, while potentially increasing the makespan, play a significant social role in creating employment opportunities.

Comparing the predictive-reactive and proactive-reactive scheduling strategies illustrates that predictive-reactive scheduling generally results in shorter makespans. This indicates that anticipating real-time events and making scheduling decisions slightly ahead of time can lead to more efficient outcomes. Furthermore, the event-driven rescheduling policy outperforms continuous rescheduling in terms of time cost, highlighting the importance of responsive rescheduling to minimize disruptions. Across various analyses, heuristic H4 consistently emerges as the most efficient algorithm. Its robustness and ability to handle dynamic scheduling scenarios make it a valuable tool for production managers seeking to optimize their processes while considering sustainability factors.

In summary, this study offers several key managerial insights. It showcases the potential of integrating sustainability criteria into real-time scheduling, providing production managers with a framework to address dynamic challenges while aligning with sustainability goals. The comparison of reformulations and heuristics demonstrates the efficiency and robustness of these methods in finding scheduling solutions. Sensitivity analysis highlights the role of critical parameters in scheduling outcomes, offering guidance for decision-making. The choice of scheduling strategy and rescheduling policy can significantly impact scheduling efficiency, and certain algorithms, such as H4, exhibit superior performance. This research equips production managers with practical tools and insights to enhance the efficiency and sustainability of their production systems.

4.7 Conclusions, findings, and future research avenues

This paper introduces a comprehensive optimization model that integrates sustainability and uncertainty considerations into a distributed permutation flow-shop scheduling problem. The main objective is to minimize the makespan while addressing constraints related to energy consumption, job opportunities, and working days lost, which have been relatively unexplored in previous studies. The model takes into account the impact of different machine operating modes on the social criteria including the number of workers and training needs on these machines. Furthermore, it incorporates three energy consumption levels according to the three states of the machines (ultra-idle, idle, and processing states) with a predefined total energy limit. To address uncertainty, real-time scheduling is employed, and various strategies and policies are evaluated such as predictive-reactive, proactive-reactive, continuous, and event-driven rescheduling approaches. By proposing efficient reformulations of the optimization model using Lagrangian relaxation and Benders decomposition, this paper offers strong solutions in comparison with the exact solver. To expedite the solution-finding process, four specific heuristics tailored to the model are employed, enhancing the overall optimization process.

We can summarize the most significant findings as follows: Figure 4.2 identified a possible solution for the proposed model based on the search space of our optimization problem. Among the various Lagrangian relaxation reformulations generated, Figure 4.3 highlighted the superior performance of LG1. The results in Figure 4.4 confirmed the high performance of the proposed BD reformulation compared to LG1. Among the four problem-specific heuristics, H4 using decision rules AF2 and NR2 was found to be the best heuristic (Figure 4.5). Then, a case study defined around the production of a flange (Figure 4.6) was used to show the applicability of the proposed approach and the efficiency of our BD reformulation (Table 4.10). Finally, the sensitivity of the makespan to the variation of the key parameters was evaluated (Figure 4.7). Finally, the real-time scheduling analysis showed that the predictive-reactive scheduling strategy is slightly better than the proactive-reactive scheduling (Table 4.12). Based on the criteria of solution quality (Figure 4.8) and CPU time (Figure 4.9), the event-driven rescheduling policy appeared to be more efficient than the continuous rescheduling policy.

Although this study contributes to the development of efficient solutions for the sustainable distributed permutation flow-shop scheduling under uncertainties, some limitations have been identified in order to recommend future works. First, we can combine real-time events with scenario-based methods for the development of a robust optimization model for the proposed problem. In addition to the maksepan, other criteria like stability of tasks assignments and tardiness can be considered to transform our model into a multi-objective optimization one. Furthermore, the use of local search metaheuristics like simulated annealing, tabu search and variable neighborhood search algorithms could improve the initial solution obtained by our heuristics. A potential future research direction is the application of an adaptive large neighborhood search to production scheduling models like the one proposed in this paper. Finally, combining our BD reformulation with some heuristic methods from an adaptive large neighborhood search can be a good idea to generate more efficient solutions for the proposed model.

CHAPTER 5

A SCENARIO-BASED ROBUST OPTIMIZATION MODEL FOR THE SUSTAINABLE DISTRIBUTED PERMUTATION FLOW-SHOP SCHEDULING PROBLEM

Amir M. Fathollahi-Fard, Lyne Woodward and Ouassima Akhrif,

Department of Electrical Engineering, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper submitted to the *Annals of Operations Research*, April 2023

Abstract

Sustainable production scheduling considers economic, environmental, and social criteria while generating the schedule of jobs in a factory. This paper formulates a Sustainable Distributed Permutation Flow-shop Scheduling Problem (SDPFSP) by considering that each machine used to process the jobs can be operated under different operating modes ranging from manual to automatic. In addition, the energy consumption as well as the number of operators required and the number of working days lost to train them have been taken into account in the proposed SDPFSP. Most importantly, this study considers multiple uncertainties including machine breakdowns, processing time, and the random arrival of new jobs. These uncertainties are formulated by a scenario-based robust optimization model as the main significant contribution of this research where the goal is to minimize the expected makespan and its deviations from probabilistic scenarios. To deal with this complex optimization problem, another innovation of this research is to propose a new metaheuristic algorithm named Adaptive Large Neighborhood Search (ALNS). The proposed algorithm uses four constructive heuristics to identify an initial solution. Then, the current solution is destroyed and repaired efficiently by the use of removal and construction heuristics to explore the search space. Thus, a local search algorithm is developed to exploit new solutions in this search space. After

implementing the proposed ALNS, an extensive computational study is provided to analyze the calibration of parameters and components of the proposed algorithm. Then, a comparison of the results with those obtained using the exact solver and state-of-the-art metaheuristics found in the literature is provided. The SDPFSP is validated through a numerical example of a flow-shop production system. Based on the results derived from our numerical example, we can conclude that our solution holds the potential to improve energy consumption by 24%, bolster job opportunities by 67%, and decrease lost workdays by 18%. Moreover, the impact of robust optimization parameters and uncertainties on optimality is investigated by performing sensitivity analysis. Finally, an in-depth discussion is provided to identify the main findings and recommendations of this research for flow-shop production systems to highlight the performance of our scenario-based robust model and the ALNS algorithm.

Keywords: Sustainable production; Flow-shop scheduling; Robust optimization; Heuristics; Large neighborhood search;

5.1 Introduction

Nowadays, there is great interest in considering all economic, environmental, and social criteria to achieve a sustainable production system while controlling uncertainties such as job processing time, machine breakdowns, and arrivals of new jobs that may be random (Varelmann et al., 2022). Since the sustainability dimensions of production systems are recently studied in the literature (Fathollahi-Fard et al., 2021), no scenario-based robust optimization approach has yet been proposed to monitor, control, and model these uncertainties in a sustainable production scheduling problem. The production system under study considers a set of factories, each comprising a set of machines. The distributed permutation flow-shop scheduling problem to be solved consists of finding an optimal sequence of jobs to be assigned to each machine that minimizes the makespan, i.e. the time that elapsed between the moment when the process of the first job is launched and the one when the last job is completed by machines among all factories (Naderi, & Ruiz, 2010). Based on the above requirements and definition, this study is the first attempt to formulate a Sustainable Distributed Permutation Flow-shop Scheduling Problem (SDPFSP) using a scenario-based robust optimization model.

SDPFSP is a challenging decision-making problem in production systems because the decisions to be made must take into account social factors such as the number of operators employed and the lost workdays as well as the amount of energy consumed which can have a significant impact on environmental pollution. In addition to assigning tasks to machines and defining their sequence, an important decision is to select the operating mode of each machine in each factory. With the development of new technologies in production systems (Dalenogare et al., 2018), each machine in each factory can be operated in different operating modes ranging from manual to automatic (Zhang et al., 2019). These operating modes may require more or less human interaction and, therefore, may affect the number of operators required, which is considered a social factor.

The selection of an operating mode has also an impact on the number of days required to train operators to operate a machine (Li, & Hsu, 2012). If a new advanced operating mode is selected to process a set of jobs, operators must learn to operate the machine in this new environment. For example, fully-automatic modes usually include programming languages as well as human machine interfaces. This requires that operators update their knowledge before they can process jobs. The number of lost workdays is defined here as the number of workdays spent teaching an operator to work with a machine using such selected operating modes.

From another point of view, SDPFSP is an extension of the energy-efficient production scheduling problem where the use of non-renewable energy and the generation of carbon emissions by machines are considered as the main sources of environmental pollution (Fathollahi-Fard et al., 2021; Marchi & Zanoni, 2017) and one of whose objectives is to achieve environmental sustainability in production systems (Corbett, & Kirsch, 2001). According to official reports, energy consumption of industrial sector accounted for approximately one-third of total U.S. energy consumption in 2017 (Conti et al. 2016). The industrial sector is expected to remain the largest energy consumer globally by 2040 (Gahm et al., 2016). Based on this prediction, the SDPFSP must take into consideration the energy consumption of machines. To do this, three different statuses are defined for the machines depending on the level of energy consumed, namely the processing status, the idle status, and the ultra-low idle status. It has been found that energy consumption for idle and ultra-low idle statuses can reach more than 40% of the total energy consumption of machines (Lu et al., 2020). The power consumption related to the processing status is the amount of power consumed by a machine while processing a job. Ultra-low idle status refers to a period when the machine is not powered while in idle status, the machine is powered and waiting to process a job. Thus, a machine in ultra-idle status consumes less energy than in idle status. Using these three statuses, the proposed SDPFSP limits the power consumption below an allowable bound.

The most important contribution of our SDPFSP is to provide a robust framework to address uncertainties such as job processing time, machine breakdowns, and random arrivals of new jobs. In this study, these events are simulated by probabilistic scenarios describing pessimistic, optimistic, and realistic cases in the production system. Thus, the goal is not only to minimize the expected makespan but also to minimize the deviations of the makespan from these possible scenarios (Mulvey et al., 1995). To this end, a robust scenario-based optimization theory (Leung et al., 2007) is deployed to define an accurate plan for production schedules while making optimal decisions for the proposed SDPFSP.

Solving the SDPFSP is computationally challenging because a large number of jobs, machines, factories, and probabilistic scenarios make the model exponentially NP-hard similar to other distributed permutation flow-shop models (Naderi, & Ruiz, 2010). Although many heuristics and metaheuristics have been applied to distributed permutation flow-shop problems (Al-Behadili, et al., 2020; Fu et al., 2019; Pan et al., 2019; Bargaoui et al., 2017; Naderi, & Ruiz, 2014), these algorithms may be unable to address specific elements of SDPFSP regarding uncertainties or sustainability criteria. To the best of our knowledge, Adaptive Large Neighborhood Search (ALNS) has not yet been applied to this type of problem in the literature. In a recent review paper, Mara et al. (2022) studied different applications of ALNS especially for routing optimization. This review paper covers 252 articles published from 2006 to 2021. In all these studies, however, there is no application for the distributed permutation flow-shop problem. This paper customizes the ALNS to make it capable of addressing specific elements of SDPFSP regarding sustainability criteria and uncertainties. Hence, one novelty is to propose a solution method based on the combination of ALNS, heuristics and local search-based algorithm of Simulated Annealing (SA) to solve the problem described above.

In conclusion, this paper adds the following contributions to the literature dealing with distributed permutation flow-shop problems:

- A robust scenario-based optimization model for the SDPFSP is formulated.
- An ALNS metaheuristic algorithm combining different removal and construction heuristics and a local search algorithm is developed to solve the SDPFSP.

The rest of this chapter is summarized as follows: Section 5.2 is an overview of relevant recent works on the distributed permutation flow-shop problem. Section 5.3 defines the problem settings and establishes a robust scenario-based optimization model for the SDPFSP. Section 5.4 presents the algorithms developed to solve the proposed SDPFSP heuristically and mathematically. Section 5.5 proposes an extensive analysis to validate the proposed ALNS metaheuristic algorithm, compares it to other similar powerful algorithms, and performs sensitivity analysis. Section 5.6 is a discussion about our contributions, practical insights, and managerial implications. Finally, a summary of this paper is provided in Section 5.6 along with a discussion of findings and recommendations for future studies.

5.2 Literature review

The distributed permutation flow-shop scheduling problem whose main goal is to minimize the makespan among many factories was modeled for the first time by Naderi & Ruiz (2010). They showed that this problem is more complex than the traditional permutation flow-shop scheduling problem where jobs are scheduled in a single factory. They defined two constructive heuristic algorithms and then improved their solutions using a Variable Neighborhood Search (VNS) metaheuristic. The distributed permutation flow-shop problem was then solved by applying different metaheuristics in the literature such as the Genetic Algorithm (GA) using local search-based operators (Gao & Chen, 2011), the iterated greedy search-based heuristic algorithm (Lin et al., 2013), the scatter search metaheuristic (Naderi & Ruiz, 2014), the chemical reaction optimization algorithm (Bargaoui et al., 2017) and a multi-neighborhood iterated greedy metaheuristic (Shao et al., 2020). Besides metaheuristics, exact algorithms like Benders decomposition (Hamzadayı, 2020) and an efficient branch-and-bound algorithm (Gmys et al., 2020) were proposed to respectively reduce the complexity of distributed permutation flow-shop scheduling problems and solve small- and medium-scale size tests optimally.

A reformulation of the distributed permutation flow-shop where the goal was to minimize the total flow-time (i.e., the total time needed to complete all jobs, as all jobs are assumed ready at time zero) instead of the makespan (i.e. the completion time of the last job) was proposed by Fernandez-Viagas et al., (2018). The proposed model was solved using different algorithms like local search-based metaheuristics (Pan et al., 2019) and a simplified neighborhood-based metaheuristic (Ruiz et al., 2019). Another modification of the distributed permutation flow-shop problem was done by Meng et al., (2019) who considered different types of products and components to be scheduled in a distributed permutation flow-shop. This new version of the problem was solved using an innovative swarm-based metaheuristic. Recently, Huang and Gu (2021) developed another variant of the distributed permutation flow-shop problem with sequence-dependent set-up times. To solve it, a novel Biogeography-Based Optimization (BBO) algorithm was developed and compared with state-of-the-art methods.

An extension of the distributed permutation flow-shop problem considers energy efficiency. In this context, Wang & Wang (2018) defined the problem to minimize the makespan and the energy consumption simultaneously. Then, they employed a knowledge-based cooperative metaheuristic to solve their problem. The energy-efficient distributed permutation flow-shop scheduling problem was also solved by different metaheuristics such as a brain storm optimization algorithm (Fu et al., 2019) and a multi-objective whale optimization algorithm (Wang et al., 2020). In addition, Han et al., (2020) formulated an energy-efficient blocking distributed permutation flow-shop problem which includes setup times. They solved it with an improved multi-objective evolutionary algorithm using VNS and local search heuristics.

As an extension to the energy-efficient distributed permutation flow-shop scheduling, the SDPFSP was firstly proposed and solved by Lu et al., (2020) using a multi-objective memetic optimization metaheuristic algorithm where they defined a penalty for processing time as a negative social factor. However, this factor is not related to the criterion for social development (Llach et al., 2015) where the number of employed operators, work injuries, and lost workdays should be taken into consideration as social indicators in a sustainable production system. In this regard, Fathollahi-Fard et al., (2021) offered a multi-objective mixed integer programming model for the SDPFSP to simultaneously optimize makespan, energy consumption, number of employed operators and lost workdays. They solved it by using a Social Engineering Optimizer (SEO) based on an adaptive search method.

Another classification of studies in this research area is based on the consideration of uncertainty in distributed permutation flow-shops. Having an efficient plan against uncertainties is an introduction to smart scheduling and production systems based on Industry 4.0 (Parente et al., 2020; Rossit et al., 2019). For example, Liu et al., (2017a) investigated a permutation flow-shop scheduling problem to find the optimal total flow-time where stochastic disruptions such as machine breakdowns and dynamic events like the random arrival of new jobs were formulated. Fu et al., (2018) minimized the total makespan and tardiness for this problem where the learning curves of operators were simulated by stochastic distribution functions. To solve this problem, a fireworks metaheuristic algorithm was developed. Framinan et al., (2019) derived real-time scheduling for a permutation flow-shop scheduling problem using a simulation-based optimization model. The key finding was the impact of using real-time information about the completion time of jobs having uncertain processing time on the ability to find an optimal rescheduling. Al-Behadili et al., (2020) used a randomized greedy search algorithm to solve another permutation flow-shop scheduling problem including machines' breakdowns and new jobs arrival. Last but not least, Jing et al., (2021) proposed an uncertain distributed permutation flow-shop problem with stochastic processing time. In this regard, a robust optimization approach was developed where a hybrid metaheuristic algorithm combining an iterated greedy search and a local search-based operator.

Based on the literature review, the following conclusions are drawn to highlight the contribution of this paper:

- The SDPFSP has only been modeled by two studies (Lu et al., 2020; Fathollahi-Fard et al., 2021), neither of which considered uncertainty;
- There has been only one robust optimization attempt to solve a distributed permutation flow-shop problem (Jing et al., 2021). However, stochastic disruptions such as machine breakdowns and the random arrival of new jobs have not been modeled by probabilistic scenarios;
- Although many metaheuristics have been reviewed in the literature of distributed permutation flow-shop, ALNS has not yet been applied or evaluated for its ability to solve DPFSP.

To address the aforementioned shortcomings, this paper proposes a robust scenario-based optimization model for SDPFSP where multiple uncertainties regarding job processing time, machine breakdowns, and random arrival of new jobs are simulated by probabilistic scenarios. The main objective is to minimize the expected makespan and its deviation through multiple stochastic and dynamic event scenarios for production planning. In the process of using an ALNS metaheuristic to minimize the objective function, four constructive heuristics and six removal heuristic are used. An initial solution is first defined using one of the four construction heuristics. Then, one of the six removal heuristics is used to destroy this solution. Again, we use one of construction heuristics to efficiently repair the destroyed solution. The algorithm iteratively selects a pair of removal-construction heuristics to explore new solutions in the search space. Finally, using a SA decision rule, a local search is performed to help our metaheuristic algorithm escape local optimal solutions.

5.3 Proposed SDPFSP

The proposed SDPFSP models a distributed permutation flow-shop scheduling problem based on sustainability criteria and uncertain parameters as a robust scenario-based optimization model to perform integrated operational planning for a set of factories. The main objective of the SDPFSP is to find an optimal expected makespan having smallest deviations in all ($s \in S$) probabilistic scenarios with respect to disruption events. The solution of the SDPFSP is an optimal sequence of ($n \in N$) jobs defined by the position ($i \in N$) of these jobs to be processed on a set of ($m \in M$) machines having a set of ($p \in P$) alternative operating modes and distributed among a set of ($f \in F$) factories. In the following, all economic, environmental, and social parameters of the proposed SDPFSP are defined. The uncertain parameters for the developed SDPFSP are then explained. Next, the scenario-based robust optimization concept is studied. Finally, the notations are introduced and the formulation of our SDPFSP is established.

5.3.1 Sustainability criteria

Sustainability criteria including economic, environmental, and social factors for the SDPFSP, are defined in this section. Regarding the economic criterion, a financial budget (B) limits the total operation cost of the machines which depends on their operating modes (CO_{mpf}) and the salaries of the assigned operators (CJ_{mpf}). Moreover, the ratio of waste (RW_{mpf}) which is affected by the operating mode selected on each machine is also taken into consideration and is limited by an upper bound (MW). The last economic criterion considered in the objective function to be minimized is the makespan which is defined as the maximum completion time of all jobs in each factory. Makespan is different from total flow-time where we compute the sum of completion times for all factories. To compute the makespan, we need to find a factory that has the maximum completion time compared to all other factories.

Regarding the environmental criterion, this study focuses on the amount of non-renewable energy consumed by the machines in order to minimize the generated carbon emissions

(Soleimani et al., 2022). To this end, the amount of energy consumed by the machines is defined according to three different statuses, namely, processing (UEC_{mpf}), idle (EC_{mpf}), and ultra-idle (IEC_{mf}) (Che et al., 2017). The total energy consumption calculated using these three statuses must be lower than a predefined upper bound ($UBEC$).

Finally, regarding the social criteria, this study considers the number of employed operators working on each machine and the lost workdays necessary for operator training (Fathollahi-Fard et al., 2021). Depending on the selected operating mode, each machine needs a specific number of workers to process the jobs (JO_{mpf}). For example, if manual operating mode is selected, a machine needs more operators than if it was operated in an automatic mode. It should be noted that from the perspective of social sustainability, it is beneficial to employ more workers in our production system (Llach et al., 2015). Therefore, the proposed model includes a lower bound (LBJ) to ensure that the optimal solution provides an acceptable number of employed workers.

In addition, this study defines lost workdays (LD_{mpf}) as the number of days needed to train an operator on a machine using a specific operating mode. Depending on the operating mode selected on a machine, the operator may need more or less knowledge to work on this machine. In this study, we considered Computer-based NC (CNC) machines that can use different automatic modes defined by the use of advanced technologies such as, for example, Programmable Logic Controllers (PLC) (Alphonsus, & Abdullah, 2016) or Automatic Position Controllers (APC) (Shilyaev et al., 2013). The duration of the training varies according to the chosen operating mode and is seen as a negative factor from an economic point of view (Llach et al., 2015). Therefore, this study defines an upper bound (UBL) for the total number of lost workdays in production planning.

5.3.2 Uncertain parameters

Compared to relevant studies (Jing et al., 2021; Al-Behadili et al., 2020; Framinan et al., 2019) in the field of DPFSP, this study addresses uncertainty differently. We not only consider the uncertainty of job processing time but also that of disruptive events such as machine breakdowns and random arrivals of new jobs in the proposed SDPFSP. For all these uncertain factors, probabilistic scenarios are adapted, each of them having a probability of occurrence (π_s). The sum of all the probabilities of occurrence is equal to one ($\sum_{s \in S} \pi_s = 1$).

As a first factor, the time required (NPC_{nmpfs}) to perform an operation O_{nmpfs}, i.e. to process a job on a machine running in a specific operating mode in a factory, depends on each scenario. This processing time is adjusted to take into account the impact of failures that may occur on the machines. In this regard, two states are defined using a binary variable (MS_{mpfs}); either a machine is available to process a job or it needs to be repaired. According to the literature (Ghaleb et al., 2020; Al-Behadili et al., 2020; Framinan et al., 2019), the time during which a machine is available to process a job (AV_{mpfs}) or the time required to repair it (RP_{mpfs}) following a failure can be estimated using exponential distribution functions. According to Ross (2019), we can assume that each machine operated in a specific mode of production has fixed failure (γ_{mp}) and repair (δ_{mp}) rates. With TF_{nmpfs} defined as the time for a failure to occur within NPC_{nmpfs} (assuming that operation O_{nmpfs} started at time 0), the job processing time was redefined as follows (Ghaleb et al., 2020):

$$PC_{nmpfs} = NPC_{nmpfs} + \left\{ \left(TF_{nmpfs} + \frac{1}{\delta_{mp}} \right) \times \left(\frac{e^{-\gamma_{mp} NPC_{nmpfs}}}{1 - e^{-\gamma_{mp} NPC_{nmpfs}}} \right) \right\} \quad \forall n \in N, m \in M, p \in P, f \in F, s \in S \tag{5.1}$$

where TF_{nmpfs} is estimated by the following formula:

$$TF_{nmpfs} = \frac{\frac{1}{\gamma_{mp}} \left(1 - e^{-\gamma_{mp} NPC_{nmpfs}}\right) - NPC_{nmpfs} e^{-\gamma_{mp} NPC_{nmpfs}}}{1 - e^{-\gamma_{mp} NPC_{nmpfs}}}, \forall n \in N, m \in M, p \in P, f \in F, s \in S \tag{5.2}$$

In addition to job processing time and machine breakdowns, this study also considers an uncertainty on machine availability for job positioning in each scenario (H_{nimpfs}). This reflects the fact that not all machines are necessarily capable of processing all jobs. Lastly, the proposed model assumes that the disruptions in our factories start from $Time_s$ at each scenario. It means that the machines are available and have no recovery time before $Time_s$. At this time, the proposed model can consider different probabilistic times of arrival of a new job which can be random to simulate this uncertain factor.

5.3.3 Scenario-based robust optimization

This study defines a robust optimization framework to handle the uncertainty in our SDPFSP. In the context of operational research, scenario-based robust optimization was firstly proposed by Mulvey et al., (1995). To introduce it, let us assume a simple optimization problem where the objective function $\varphi = f(x)$ is to be minimized. To manage the uncertainty, we transform the variable x to x_s which depends on scenario *s*. Thus, for each scenario *s*, a cost φ_s is defined:

$$\varphi_s = f(x_s) \qquad \forall s \in S \tag{5.3}$$

Consequently, the uncertain problem aims to minimize the total cost (*Z*) of all scenarios as defined hereafter:

$$min\ Z = \sum_{s \in S} \pi_s \times \varphi_s \tag{5.4}$$

where π_s is the probability of occurrence of each scenario *s*.

Mulvey et al., (1995) updated this scenario-based stochastic programming model to consider the deviation of each scenario from their expected value as follows:

$$min\ Z = \sum_{s \in S} \pi_s \times \varphi_s + \lambda \sum_{s \in S} \pi_s (\varphi_s - \sum_{\substack{s' \in S \\ s' \neq s}} \pi_{s'} \times \varphi_{s'})^2 \tag{5.5}$$

where λ is a weighting factor chosen between zero and one. The first term of Eq. (5.5) is similar to that shown in Eq. (5.4). The additional term in Eq. (5.5) describes the deviation of each scenario from other scenarios by computing the variance of all scenarios. Mulvey et al., (1995) called the formulation of Eq. (5.5), a robust programming model. Although this formulation may provide a better solution than that of the stochastic programming described in Eq. (5.4), its right- term makes the objective function non-linear. Leung et al., (2007) solved this issue by linearizing Eq. (5.5) as follows:

$$min\ Z = \sum_{s \in S} \pi_s \times \varphi_s + \lambda \left(\sum_{s \in S} \pi_s \times \varphi_s - \sum_{\substack{s' \in S \\ s' \neq s}} \pi_{s'} \times \varphi_{s'} + 2\theta_s \right) \tag{5.6}$$

where θ_s is an auxiliary variable supported by the following constraint set:

$$\varphi_s - \sum_{\substack{s' \in S \\ s' \neq s}} \pi_{s'} \times \varphi_{s'} + \theta_s \geq 0 \qquad \forall s \in S \tag{5.7}$$

In this study, we applied the concept of scenario-based robust optimization described by Eqs. (5.6) and (5.7) to our SDPFSP. The proposed model can generate several probabilistic scenarios to address the previously listed uncertain factors while establishing a robust plan against uncertainties.

5.3.4 Notations and problem formulation

Based on the definition of our SDPFSP including sustainability criteria, uncertainty factors, and scenario-based robust optimization, the following notations are used to describe the proposed model.

Sets:

$f \in F$	Set of factories,
$m \in M$	Set of machines,
$n \in N$	Set of jobs,
$p \in P$	Set of operating modes,
$i \in N$	Set of job's positions in a schedule,
$s \in S$	Set of probabilistic scenarios for disruptive events,

Parameters:

B	Maximum budget allowed for the installation of machines and operating modes as well as for the salary of operators (in \$),
CO_{mpf}	Training cost for implementing operating mode *p* on machine *m* in factory *f* (in \$),
JO_{mpf}	Number of employed operators needed to operate machine *m* of factory *f* in operating mode *p,*
CJ_{mpf}	Hourly wage of operators running machine *m* in operating mode *p* in factory *f* (in \$/h),
LD_{mpf}	Number of days needed for training an operator to work on machine *m* using operating mode *p* in factory *f,*
MW	Maximum allowable total waste ratio on all machines of all factories,
RW_{mpf}	Waste ratio of machine *m* using operating mode *p* in factory *f,*
O_{nmpfs}	Operation of job *n* on machine *m* using operating mode *p* in factory *f* under scenario *s,*
PC_{nmpfs}	Processing time of operation O_{nmpfs} including machines' breakdowns (in *h*)
UEC_{mpf}	Useful energy consumed by machine *m* of factory *f* operated in mode *p* while being in process status (in *kWh),*
EC_{mpf}	Energy consumed by machine *m* of factory *f* operated in mode *p* while being in idle status (in *kWh),*
IEC_{mf}	Energy consumed by machine *m* of factory *f* while being in ultra-idle status (in *kWh),*
$UBEC$	Maximum total energy consumption allowed (in *kWh),*
LBJ	Minimum total number of employed operators allowed,
UBL	Maximum total number of lost workdays allowed,
π_s	Probability of occurrence of scenario *s*,
$Time_s$	Time in scenario *s* where the random arrival of a new job is estimated to occur (in *h*).
MS_{mpfs}	Availability of machine *m* located in factory *f* and using operating mode *p* in scenario *s*; equals to 1 if the machine is available to process jobs; otherwise, 0.

AV_{mpfs} Time where machine m located in factory f and using operating mode p in scenario s is available to process a job. This is a positive value (in h) if the machine is available to process jobs ($MS_{mpfs} = 1$); otherwise, 0.

RP_{mpfs} Time where machine m located in factory f and using operating mode p in scenario s is unavailable since it must be repaired. It is a positive value (in h) if the machine must be repaired ($MS_{mpfs} = 0$); otherwise, 0.

H_{nimpfs} If job n can be processed at position i on machine m located in factory f and using operating mode p in scenario s, 1; otherwise, 0.

λ Robust coefficient parameter.

Decision variables:

Y_{mpf} If operating mode p is selected for machine m in factory f, 1; otherwise, 0.

ST_{impfs} Starting time (in h) of the job at position i in the planned sequence to be assigned to machine m of factory f using operating mode p in scenario s,

X_{nimpfs} If job n is scheduled at position i in the planned sequence to be assigned to machine m of factory f using operating mode p in scenario s, 1; otherwise, 0.

A_{fs} Number of jobs assigned to factory f in scenario s. This is an auxiliary variable dependent on X_{nimpfs}.

C_{impfs} Completion time (in h) of a job at position i in the planned sequence to be assigned to machine m in factory f using operating mode p in scenario s. This is an auxiliary variable dependent on X_{nimpfs} and ST_{impfs}.

CT_{fs} Time (in h) required to complete all jobs in factory f according to scenario s. This is an auxiliary variable dependent on ST_{impfs}.

$CMAX_s$ Total makespan (in h) in scenario s. This is an auxiliary variable depending on CT_{fs}.

θ_s Auxiliary variable used in the definition of the expected makespan.

Z Total expected makespan (in h) computed according to probabilities of occurrence, $CMAX_s$ and θ_s.

Our scenario-based robust optimization model defining the proposed SDPFSP is described as follows:

$$min \qquad Z = \sum_{s \in S} CMAX_s \times \pi_s + \lambda(\sum_{s \in S} (\pi_s \times CMAX_s - \sum_{\substack{s' \in S \\ s \neq s'}} \pi_{s'} \times CMAX_{s'} + 2\theta_s)) \tag{5.8}$$

$$s.t.$$

$$\sum_{m \in M} \sum_{p \in P} \sum_{f \in F} (Y_{mpf} \times JO_{mpf} \times CJ_{mpf}) + \sum_{m \in M} \sum_{p \in P} \sum_{f \in F} (Y_{mpf} \times CO_{mpf}) \leq B \tag{5.9}$$

$$\sum_{m \in M} \sum_{p \in P} \sum_{f \in F} (Y_{mpf} \times RW_{mpf}) \leq MW \tag{5.10}$$

$$\sum_{i \in N} \sum_{f \in F} X_{nimpfs} = 1, \qquad \forall n \in N, m \in M, p \in P, s \in S \tag{5.11}$$

$$\sum_{n \in N} \sum_{f \in F} X_{nimpfs} = 1, \qquad \forall i \in N, m \in M, p \in P, s \in S \tag{5.12}$$

$$\sum_{m \in M} \sum_{p \in P} \sum_{n \in N} \sum_{i \in N} X_{nimpfs} = A_{fs}, \qquad \forall f \in F, s \in S \tag{5.13}$$

$$\sum_{i \in N} \sum_{n \in N} X_{nimpfs} \leq N \times Y_{mpf}, \qquad \forall m \in M, p \in P, f \in F, s \in S \tag{5.14}$$

$$\sum_{p \in P} Y_{mpf} = 1, \qquad \forall m \in M, f \in F \tag{5.15}$$

$$X_{nimpfs} \leq H_{nimpfs} \qquad \forall i, n \in N, m \in M, p \in P, f \in F, s \in S \tag{5.16}$$

$$ST_{impfs} \geq \sum_{n \in N} X_{nimpfs} \times \{MS_{mpfs} AV_{mpfs} + (1 - MS_{mpfs}) RP_{mpfs}\}), \tag{5.17}$$

$$\forall i \in N, m \in M, p \in P, f \in F, s \in S$$

$$C_{impfs} \geq ST_{i,m-1,pfs} + \sum_{n \in N} X_{nimpfs} \times PC_{nmpfs}, \tag{5.18}$$

$$\forall i \in N, m > 1, p \in P, f \in F, s \in S$$

$$C_{impfs} \geq ST_{i-1,mpfs} + \sum_{n \in N} X_{nimpfs} \times PC_{nmpfs}, \tag{5.19}$$

$$\forall i > 1, m \in M, p \in P, f \in F, s \in S$$

$$CT_{fs} \geq \sum_{i \in N} \sum_{m \in M} \sum_{p \in P} C_{impfs}, \qquad \forall f \in F, s \in S \tag{5.20}$$

$$\sum_{m \in M} \sum_{p \in P} \sum_{f \in F} (Y_{mpf} \times EC_{mpf}) \tag{5.21}$$

$$+ \sum_{m \in M} \sum_{p \in P} \sum_{f \in F} \sum_{n \in N} \sum_{s \in S} \pi_s \left(UEC_{mpf} \times Y_{mpf} \times PC_{nmpfs} \right)$$

$$+ \sum_{m \in M} \sum_{f \in F} IEC_{mf} \times \sum_{p \in P} Y_{mpf} \leq UBEC$$

$$\sum_{m \in M} \sum_{p \in P} \sum_{f \in F} (Y_{mpf} \times JO_{mpf}) \geq LBJ \tag{5.22}$$

$$\sum_{m \in M} \sum_{p \in P} \sum_{f \in F} (Y_{mpf} \times LD_{mpf}) \leq UBL \tag{5.23}$$

$$CMAX_s \geq CT_{fs} \qquad \forall f \in F, s \in S \tag{5.24}$$

$$CMAX_s - \sum_{\substack{s' \in S \\ s' \neq s}} \pi_{s'} CMAX_{s'} + \theta_s \geq 0 \qquad \forall s \in S \tag{5.25}$$

$$A_{fs}, ST_{impfs}, C_{impfs}, CT_{fs}, CMAX_s, \theta_s \geq 0 \tag{5.26}$$

$$Y_{mpf}, X_{nimpfs} \in \{1,0\} \tag{5.27}$$

The objective function defined in Eq. (5.8) is limited by constraints (5.9) to (5.25) while Eqs. (5.26) to (5.27) define the manipulated variables. The objective function aims to minimize the expected makespan as well as the deviation of other scenarios from this expected makespan according to the concept of scenario-based robust optimization described in Eqs (5.3) to (5.7).

Constraint (5.9) guarantees that the total cost of machines and the salary of operators do not exceed the predefined budget. Constraint (5.10) ensures that the total ratio of waste according to each selected operating mode remains lower than the maximum ratio of allowed wastes. Constraints (5.11) and (5.12) assign each job to a position in the schedule. Constraint set (5.13) computes the number of jobs assigned to each factory while constraint set (5.14) ensures that a machine process all the assigned jobs using a single operating mode within its scheduled plan. Thus, the constraint set (5.15) guarantees that only one operating mode is selected on each machine in each factory during the schedule. Constraint set (5.16) verifies the possibility for the machines to process jobs. Constraint set (5.17) takes into account delays caused by disruptive events such as random arrivals of new jobs and machine breakdowns in the computation of starting times of jobs in the schedule. Constraints (5.18) and (5.19) compute the starting time of each job to be processed by a machine. The time required to complete all the jobs of each factory in each scenario is computed by the constraint set (5.20). In the proposed model, we use inequality sign instead of equality sign for this constraint set to help the exact solver for analyzing the flexibility of this constraint set. Since it is a minimization problem, the completion time at each factory will not get a greater value than the completion time of jobs at each factory under each scenario.

Based on the sustainability criteria, the total energy consumption computed according to the three statuses defined earlier (processing, idle, and ultra-idle statuses) is limited by constraint (5.21). Constraint (5.22) guarantees that minimum number of operators required regarding the social criterion is met. As such, the number of lost workdays according to the selected operating modes is calculated in constraint (5.23). For all factories in each scenario, the makespan is defined by constraint set (5.24) to find the maximum completion time among all factories. To keep our model linear, we use inequality sign instead of maximum function to ensure that only the maximum value of the completion time is considered for the makespan since it is a minimization problem. The deviation of makespan from each scenario is computed in constraint set (5.25). Note that if the makespan in a scenario is bigger than the expected value, θ_s will get zero value due to the minimization problem. Finally, non-binary and binary decision variables are respectively defined in Eqs. (5.26) and (5.27).

5.4 Proposed ALNS algorithm

To solve the optimization problem presented in Section 5.3.4, an Adaptive Large Neighborhood Search (ALNS) using six removal and four construction heuristics along with a local search-based algorithm and a Simulated Annealing (SA) metaheuristic is chosen. Large Neighborhood Search (LNS) was first introduced by Shaw (1998). The main idea of LNS is to destroy the current solution and reconstruct it in order to improve diversity and avoid convergence towards a local solution (Schrimpf et al., 2000). Similar to other metaheuristics, this algorithm alternates between exploitation and exploration phases to find high-quality solutions. In the exploitation phase, based on heuristics, the algorithm can exploit the information contained in a solution to generate a new one. The number of removal-construction heuristics helps the algorithm to explore optimal solutions in the exploration phase. The main difference between ALNS and the original LNS is that the former uses an adaptive strategy to select a pair of removal and construction heuristics based on the roulette wheel selection algorithm. For construction heuristics, this study focuses on four decision rules which are an extension of the rules proposed by Naderi and Ruiz (2010). However, the removal heuristics proposed and used in this paper are new to this research area.

A general view of our ALNS metaheuristic algorithm is shown in Figure 5.1. The algorithm begins by creating an initial solution using our construction heuristics. Then, a pair of removal-construction heuristics is selected to destroy the current best-known solution first before repairing it in order to explore a new feasible solution. Based on a SA decision rule, the algorithm decides whether to accept or reject this new solution. In addition, the weights of removal and construction heuristics are updated based on the performance record for generating this new solution. Next, in a sub-iteration, our ALNS performs a local search to update and improve the current best-known solution. Once the maximum number of iterations has been reached, the algorithm displays the best-known final solution.

In the following subsections, we first present the solution representation and the search space of our metaheuristic algorithm (Section 5.4.1). The four construction heuristics used to develop an initial solution are then explained (Section 5.4.2). Then, our removal heuristics are defined and illustrated mathematically (Section 5.4.3). The local search used at the end of each iteration, is presented as a sub-loop of our metaheuristic algorithm (Section 5.4.4). The procedure for selecting each pair of removal and construction heuristics is then studied (Section 5.4.5). Finally, the complete algorithmic framework of the proposed ALNS metaheuristic algorithm is explained (Section 5.4.6).

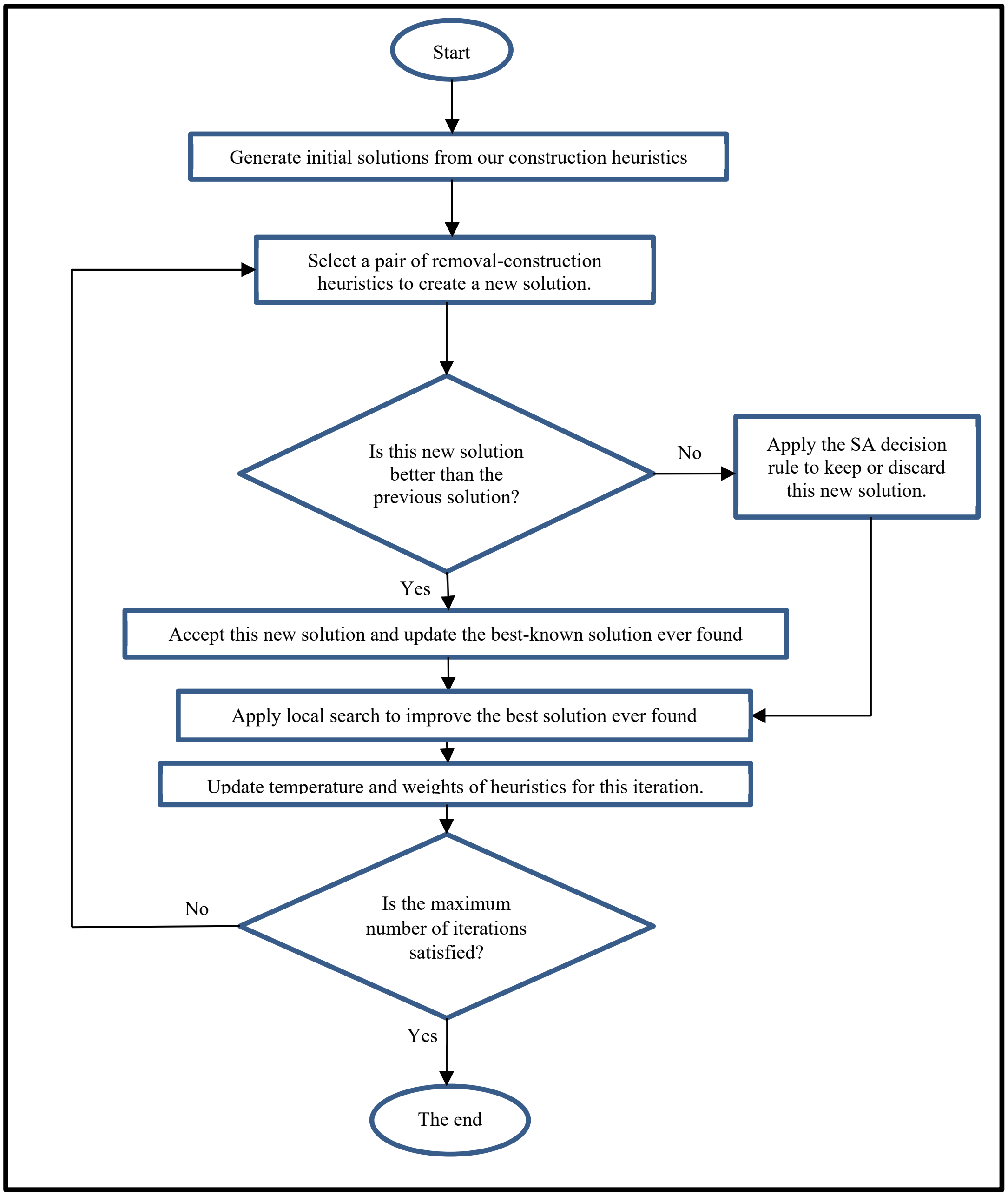


Figure 5.1 Flowchart of the proposed ALNS metaheuristic algorithm

5.4.1 Search space and solution representation

In the proposed ALNS, the search space is defined as a set of feasible solutions where the global optimum (Sol^*) is expected to be found. At each iteration, the metaheuristic selects a solution among these feasible solutions (Mara et al., 2022). This selection is made intelligently since by using different decision rules, our heuristics guide the ALNS algorithm in the exploration of new high-quality solutions. However, this procedure also has a random aspect with regard to the selection of removal and construction heuristics. All the solutions including different alternatives with respect to the selected operating modes and the jobs assigned to the machines in each scenario (using the design vector X_{nimpfs}) define the search space of our ALNS. In this respect, the representation of the solution based on the designation of X_{nimpfs} has three parts, namely, the selection of the operating modes of each machine, the assignment of jobs to these machines, and the sequence according to which the jobs will be processed depending on the availability of the machines as described by the set of constraints (5.16).

The proposed ALNS metaheuristic algorithm explores and exploits new neighborhoods from the search space defined by the main design variable ($i.e., X_{nimpfs}$). At each iteration ($it \in \{1, 2, \dots, MaxIt\}$), neighborhoods are found in the search space by using a pair of removal and construction heuristics. In this regard, it is first necessary to select the operating modes (Y_{mpf}) then determine the sequence of jobs to be carried out on the machines in each scenario (X_{nimpfs}). Other decision variables are computed by constraints (5.13), (5.14), (5.17), (5.18), (5.19), (5.20), (5.24) and (5.25). To illustrate how to define a solution in the search space, consider the following example where there are two factories (F_1, F_2), four machines ($M_1, M_2,$ located in F_1 and M_3, M_4 located in F_2), each of them having two operating modes (1-automatic and 2-manual), two scenarios (S_1, S_2), and a total of 10 jobs to be processed. Figure 5.2 shows the solution representation for this example. Firstly, Figure 5.2(a) shows the selection of operating modes on each machine. In this example, machines M_1 and M_4 operate in manual mode while machines M_2 and M_4 operate in automatic mode. Figure 5.2(b) and Figure 5.2(c) show the sequence of jobs planned according to the first (S_1) and second (S_2) scenarios, respectively. The time required to complete all jobs in a factory under each scenario is also shown of Figure 5.2(b) and Figure 5.2(c). For each factory, there is a time ($Time_s$) where an uncertain event may occur. Therefore, each sequence of jobs in each scenario is divided into two cases, namely, jobs to be processed before $Time_s$ and after $Time_s$. More details on the design of these solutions are provided in the next sub-section.

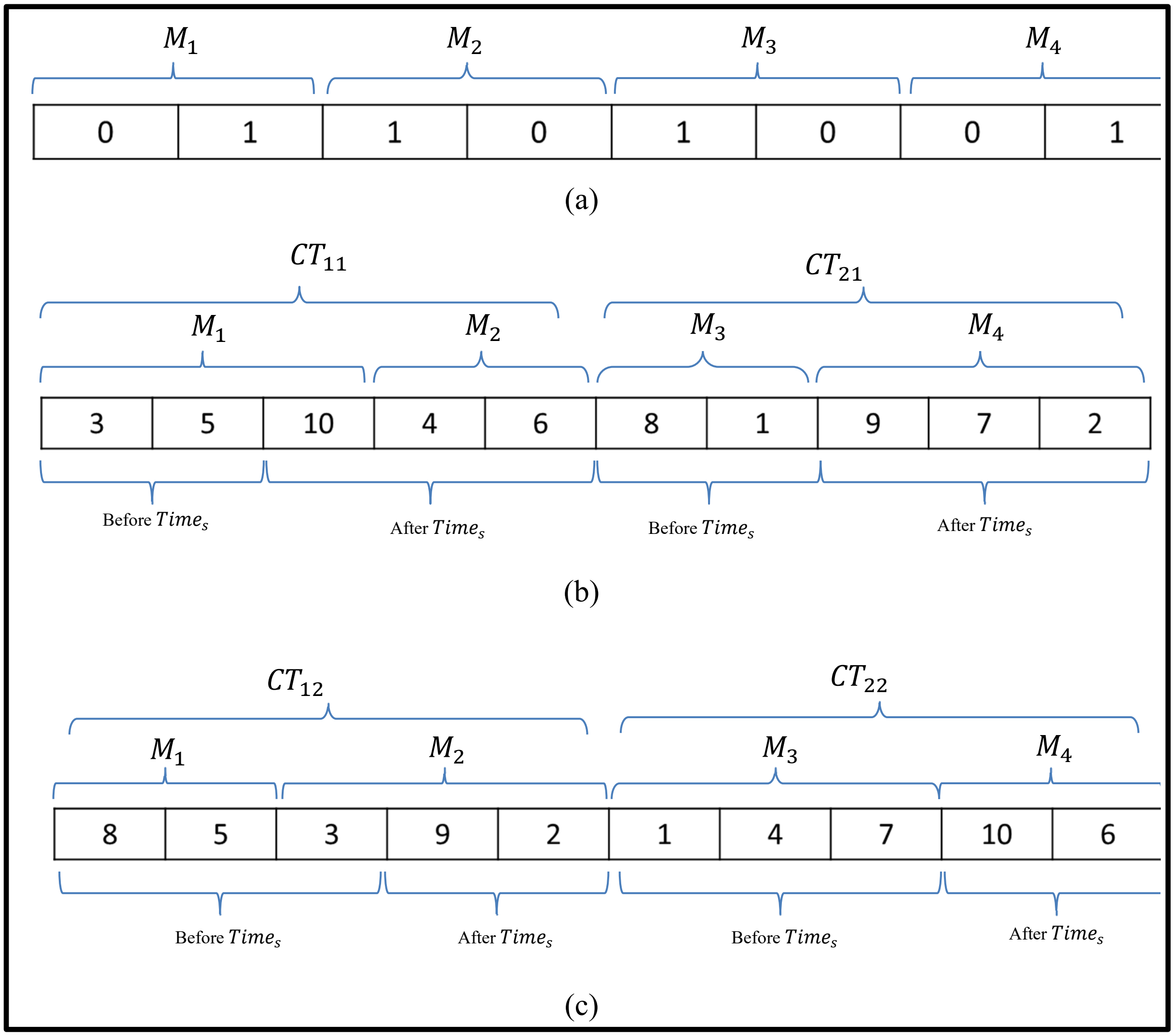


Figure 5.2 Solution representation, i.e., (a) assignment of operating modes, (b) sequence of jobs according to scenario S_1, (c) sequence of jobs according to scenario S_2

5.4.2 Construction and reconstruction of initial and destroyed solutions

To construct an initial solution (Sol^0), we have considered four heuristics. This initial solution is in fact the best-known current solution (Sol^*) at the first iteration. The assignment of operating modes (Figure 5.2(a)) is not related to probabilistic scenarios as is the case for the allocation and sequencing of jobs on the machines (Figure 5.2(b) and Figure 5.2(c)). Thus, the first step in finding a solution is to select the operating modes (Y_{mpf}). Then, in each probabilistic scenario, our heuristics follow a custom decision rule before the disruptive event

($Time_s$) and then employ another decision rule after this disruptive event to finalize the scheduling of all jobs. It should be noted that we assume that the proposed problem always has a set of feasible solutions where our heuristic algorithms aim to find the best one within a fixed number of iterations.

The following steps are performed in all the construction heuristics to select the operating modes of each machine:

- **Step 0**: For each operating mode, each machine, and each factory, we compute the average processing time of jobs ($\sum_{s \in S}(\sum_{n \in N} PC_{nmpfs} / N) / S$).
- **Step 1**: For each machine, we select the operating mode (Y_{mpf}) leading to the lowest average processing time which can be found in the matrix calculated at **Step 0**.
- **Step 2**: If the initial assignment of operating modes based on the average job processing does not satisfy one or more of the constraints (5.9), (5.10), (5.21), (5.22) and (5.23), go to the next step. Otherwise, go to **Step 3**.
- **Step 2.1**: If constraint (5.9) is not satisfied, identify the machine having the highest implementation cost (CO_{mpf}). Change the selection of the operating mode for this machine and see if the implementation cost is reduced. If so, keep this selection. If not, switch to the mode of operation previous selected for this machine. Repeat this step for the machine with the highest implementation cost among those not yet tested until constraint (5.9) becomes feasible.
- **Step 2.2**: If constraint (5.10) is not satisfied, identify the machine having the highest error ratio (RW_{mpf}). Change the selection of the operating mode for this machine and see if the error rate is reduced. If so, keep this selection. If not, switch to the mode of operation previously selected for this machine. Repeat this step for the machine with the highest error rate among those not yet tested until constraint (5.10) becomes feasible
- **Step 2.3**: If constraint (5.21) is not satisfied, identify the machine with the highest total energy consumption ($UEC_{mpf} + EC_{mpf} + IEC_{mf}$). Change the selection of the operating mode for this machine and see if the total energy consumption is reduced. If so, keep this selection. If not, switch to the mode of production previously selected for

this machine. Repeat this step for the machine with the highest total energy consumption among those not yet tested until constraint (5.21) becomes feasible.

- **Step 2.4**: If constraint (5.22) is not satisfied, identify the machine requiring the smallest number of employed workers (JO_{mpf}). Change the selection of the operating mode for this machine and see if the number of employed workers is increased. If so, keep this selection. If not, switch to the mode of production previously selected for this machine. Repeat this step for the machine with the smallest number or employed workers among those not yet tested until constraint (5.22) becomes feasible.
- **Step 2.5**: If constraint (5.23) is not satisfied, identify the machine leading to the largest number of lost workdays (LD_{mpf}). Change the selection of the operating mode for this machine and see if the number of lost working days is reduced. If so, keep this selection if not, switch to the mode of production previously selected for this machine. Repeat this step for the machine with the highest number of lost working days among those not yet tested until constraint (5.23) becomes feasible.
- **Step 3**: Once all these constraints have been verified, if an infeasibility remains, go back to its relevant step to update the assignment of operating modes and makes the solution feasible. Then, update X_{nimpfs} using constraint set (5.14) and continue applying decision rules to assign jobs to machines and to determine their positions in the sequence.

One of the following two decision rules is applied to assign and sequence jobs before $Time_s$, the time at which a disruptive event occurs which varies in each scenario. Before applying these decision rules, we need to assign the first job to initialize the makespan calculation. Once the makespan is less than the disruptive event $Time_s$, one of the two following decision rules, i.e., NR1 and NR2 defined by Naderi & Ruiz (2010) and customized for our SDPFSP, is applied:

- **NR1:** Assign job *n* to the factory having the minimum completion time before this assignment. If there is more than one factory with the minimum completion time, select the first one. Then, a machine that is not busy processing an operation in this factory is selected. After assigning a job to a factory, change its processing time to an infinite value to avoid selecting it in the next assignment.

- **NR2:** Assign job n to the factory having the minimum completion time after this assignment. If there is more than one factory with the minimum completion time, select the first one. Then, a machine that is not busy processing an operation in this factory is selected. After the assignment of a job to a factory, change its processing time to an infinite value to avoid selecting it in the next assignment.

However, NR1 and NR2 decision rules are unable to manage the uncertainty related to machine breakdowns that occur after $Time_s$ in each scenario and require recovery time. In this regard, NR1 and NR2 are extended to rules AF1 and AF2 which take into account the recovery time (RP_{mpfs}) needed after a machine breakdown occurs according to the set of constraints (5.17). These decision rules are defined hereafter:

- **AF1:** Identify the factory having the minimum completion time before the job assignment to be performed. If there is more than one factory with the minimum completion time, select the first one. Next, identify the machine of this factory that has the minimum failure recovery time. Again, if there is more than one machine with the minimum failure recovery time, select the first one. Finally, assign the job having the lowest processing time to this machine and change its processing time to an infinite value to prevent it being selected during the next assignment.
- **AF2:** Except for the very first job assignment where the factory with the maximum completion time (instead of the minimum) is selected, rule AF2 follows exactly the same procedure as rule AF1 for job assignments. Note that AF2 finds the minimum completion time after the job is assigned.

Overall, the proposed approach first applies Steps 0 to 3 before using one of the four construction heuristics (C1, C2, C3 and C4) to identify a feasible solution. Each of these construction heuristics uses different rules described hereafter:

- **C1:** Apply NR1 to assign jobs (X_{nimpfs}) up to $Time_s$ and AF1 to assign the remaining jobs.
- **C2:** Apply NR2 to assign jobs (X_{nimpfs}) up to $Time_s$ and AF1 to assign the remaining jobs.
- **C3:** Apply NR1 to assign jobs (X_{nimpfs}) up to $Time_s$ and AF2 to assign the remaining jobs.
- **C4:** Apply NR2 to assign jobs (X_{nimpfs}) up to $Time_s$ and AF2 to assign the remaining jobs.

Thus, for each machine in each factory, an operating mode (Y_{mpf}) is first selected using Steps 0 to 3. Then, for each scenario, a construction heuristic is selected. Consequently, rule NR1 or rule NR2 is applied to assign the jobs and specify their sequence (X_{nimpfs}) up to $Time_s$ where a disruptive event occurs. After this disruptive event, rule AF1 or rule AF2 is used to assign the remaining jobs and specify their sequence. Finally, the makespan of each scenario ($CMAX_s$) is computed and the value of the objective function (*Z*) is obtained.

In order to find the initial solution (Sol^0), we run the construction heuristics individually and then select the best solution which is identified as the best-known current solution (Sol^*) at the first iteration ($it = 1$). The construction heuristics will also be used to repair a solution destroyed after the application of a removal heuristic to improve diversity. These removal heuristics will now be defined.

5.4.3 Removal heuristics

An important feature of our ALNS is its propensity to promote good diversity by the use of a set of six removal heuristics. Indeed, if construction heuristics can be linked to the capacity of exploitation, the removal heuristics contribute to explore new regions in the search space (Mara et al., 2022). Since ALNS has not yet been applied to SDPFSP in the literature (Fathollahi-Fard et al., 2021), there exist no removal heuristics that meet the requirements of the problem under study. Thus, six SDPFSP-specific removal heuristics are proposed to improve diversity by destroying the current solution at each iteration.

Consider $it \in \{1, 2, \dots, MaxIt\}$ as the iteration index of the ALNS algorithm. At the very first iteration (*it* = *1*), the current best solution is taken from the solutions generated by the construction heuristics C1, C2, C3 and C4, i.e., $Sol^* = Sol^0$. Then, at each of the following iterations, elements are removed from the solution using one of the removal heuristics. Thus, some values of the decision variables X_{nimpfs}^{it} and Y_{mpf}^{it} are modified to generate a tentative solution for the next iteration, i.e., new values for X_{nimpfs}^{it+1} and Y_{mpf}^{it+1}.

The set of six removal heuristics includes a random-machine-based removal heuristic (R1), a random-operating-mode-based removal heuristic (R2), a random-job-based removal heuristic (R3), a maximum-makespan-machine-based removal heuristic (R4), a low-use-machine-based removal heuristic (R5), and a high-processing-time-based removal heuristic (R6). In all these removal heuristics, a parameter (Q) ranging from zero to one allows to choose the percentage of elements to be removed from the solution at iteration *it* (Sol^{it}). With the exception of R2, all removal heuristics modify the sequence of jobs in each scenario (Figure 5.2(b) and Figure 5.2(c)). The R2 heuristic however aims to change the selected operating modes of each machine (Figure 5.2(a)). For each removal heuristic, the way to obtain partial feasible solutions is different as described hereafter:

- **Random-machine-based removal heuristic (R1):** From set *M*, randomly select a maximum percentage (Q) of machines. Remove the jobs that have been assigned to the selected machines.
- **Random-operation-mode-based removal heuristic (R2):** From set *M*, randomly select a maximum percentage (Q) of machines. Remove the operating modes that have been chosen for these selected machines.
- **Random-job-based removal heuristic (R3):** From set *N*, randomly select a percentage (Q) of jobs and remove them from the sequence of jobs assigned to machines.

- **Maximum-makespan-machine-based removal heuristic (R4):** Select the factory having the maximum completion time of jobs. For this factory, select a random percentage (Q) of machines and remove jobs from the job sequence assigned to the selected machines. Moreover, remove a percentage $(Q/2)$ of jobs from job sequences assigned to machines in other factories at random.
- **Low-use-machine-based removal heuristic (R5):** Select the least-use factory that is, the one with the minimum job completion time. Select a random percentage (Q) of machines in this factory and remove the sequences of jobs that have been assigned to these machines. Moreover, remove a percentage $(Q/2)$ of jobs from job sequences assigned to machines in other factories at random.
- **High-processing-time-based removal heuristic (R6):** Sort factories in descending order of completion times and select the first Q percent of factories in this sorted list. Then, in each selected factory, sort the jobs in descending order of processing times. Remove the first (Q) percent of these sorted jobs from the job sequence assigned to machines in these factories.

5.4.4 Local search algorithm

As mentioned earlier, at each iteration, the proposed ALNS algorithm performs a local search to slightly modify the best solution ever found so far using a pair of removal-construction heuristics. In this regard, there is a maximum number of sub-iterations (*SubIt*) in the main loop of the algorithm making it possible to exploit new solutions in order to escape the local optimal solutions identified by the pair of removal-construction heuristics. The local search begins with the random selection of one or two machines for which jobs and operating modes have been removed from the sequences which were initially assigned to them. Then, one of the four construction heuristics is randomly applied to repair this partial feasible solution. Another random solution from the assignment of operating modes and job scheduling, is also created. The best solution among these new solutions is compared to the best solution ever found so far (Sol^*). Thus, the local search algorithm can be defined by the following steps:

- **Step 0**: Consider the current best solution (Sol^{*}) as the input of this local search algorithm.
- **Step 1**: Select one or two machines randomly from this solution and remove the jobs and operating modes that were assigned to them.
- **Step 2:** Repair this partial feasible solution using one of construction heuristics C1 to C4 to repair the solution.
- **Step 3:** In addition to this repaired solution, create a random solution from the search space (defined in Section 5.4.1) and select the best solution among these 2 solutions.
- **Step 4:** If this new best solution (Sol^{it}) is better than the best solution ever found so far (Sol^{*}), update the latter.
- **Step 5:** If the maximum number of sub-iterations (*SubIt*) has not been reached, go back to **Step 1**. Otherwise, display the current best solution ever found so far.

5.4.5 Selection procedure of the pair of the removal-construction heuristics

At each iteration of the proposed metaheuristic algorithm, a pair of construction and removal heuristics is applied to obtain a new solution. This choice is made using the roulette wheel selection where the selection probabilities of each heuristic are based on their respective adaptive weights.

At the first iteration, the four weights (iw_c^{+} where $c \in C$) associated with the construction heuristics and the six weights (rw_r^{-} where $r \in R$) associated with the removal heuristics are all equal to one. At each subsequent iteration, the weights are updated using the following equations:

$$rw_r^{-} = \vartheta \times rw_r^{-} + (1-\vartheta)\Omega \qquad \forall r \in R \tag{5.28}$$

$$iw_c^{+} = \vartheta \times iw_c^{+} + (1-\vartheta)\Omega \qquad \forall c \in C \tag{5.29}$$

where $\vartheta \in [0,1]$ is a forgetting factor and Ω is a score given to each heuristic according to its own recorded performance in finding the best solution in the previous iteration, as defined hereafter:

$$\Omega = \begin{cases} \varpi_1 & \text{If the new solution } (Sol^{it}) \text{ is the new global best-known solution } (Sol^{*}). \\ \varpi_2 & \text{If the new solution is accepted.} \\ \varpi_3 & \text{If the new solution is rejected.} \end{cases} \tag{5.30}$$

Three possible scores ϖ_1, ϖ_2 and ϖ_3 are set such that $\varpi_1 \geq \varpi_2 \geq \varpi_3$. Once the weights are updated, the probability of each removal (P_r^{rw}) and construction (P_c^{iw}) heuristic in the roulette wheel selection is computed as follows:

$$P_r^{rw} = \frac{rw_r^-}{\sum_{r \in R} rw_r^-}, \qquad \forall r \in R \tag{5.31}$$

$$P_c^{iw} = \frac{iw_c^+}{\sum_{c \in C} iw_c^+}, \qquad \forall c \in C \tag{5.32}$$

To illustrate the roulette wheel selection mechanism (Abreu et al., 2020), let us define a numerical example. After few iterations, assume that the weights of construction heuristics (C1 to C4) are 1.45, 1, 0.85, 0.75 respectively. Using Eq. (5-32), their probabilities are 0.36, 0.25, 0.21 and 0.19 respectively. By generating a random continuous variable between zero and one (*rand)*, one of the 4 construction heuristics will be selected according to the roulette wheel function, $f(rand)$ which is defined as follows:

$$f_{(rand)} = \begin{cases} C1 & \text{if } 0 \leq rand < 0.36 \\ C2 & \text{if } 0.36 \leq rand < 0.6 \\ C3 & \text{if } 0.6 \leq rand < 0.81 \\ C4 & \text{if } 0.81 \leq rand \leq 1 \end{cases} \tag{5.33}$$

5.4.6 Main steps of the proposed ALNS algorithm

As mentioned earlier, our ALNS algorithm starts with an initial solution (Sol^0) which is the best solution among the solutions found by the 4 constructive heuristics. The solution to be destroyed and then repaired by the pair of removal and construction heuristics, is denoted

Sol^{it}. Once the pair of selected removal-construction heuristics is applied, a new solution (New_Sol^{it}) is found. If this solution is better than the best solution ever found so far (Sol^{*}), the latter is updated and, in the next iteration the algorithm considers this solution as an input for the next pair of removal-construction heuristics to select. Otherwise, using a decision rule from SA algorithm, (New_Sol^{it}) may or may not be accepted as an input for the next iteration instead of Sol^{it}. If rejected, the current solution (Sol^{it}) will be used as input for the next iteration (Sol^{it+1}). The decision to accept or reject New_Sol^{it} is based on comparing a randomly generated continuous number between zero and one ($rand_1$) and a probability index p which is calculated as follows:

$$p = e^{-\Delta/Tem} \qquad \text{where} \quad \Delta = \left|Z(New\_Sol^{it}) - Z(Sol^{*})\right| \tag{5.34}$$

where $Z(New\_Sol^{it})$ and $Z(Sol^{*})$ are computed using Eq. (5.8).

At each iteration *it*, the current temperature (*Tem*) will be updated as follows:

$$Tem = redu \times Tem \tag{5.35}$$

where $redu$ is the temperature damping factor. It should be noted that this decision rule was taken from the SA metaheuristic algorithm (Bellio et al., 2021).

At each iteration, our ALNS metaheuristic algorithm has a sub-loop (itself having a maximum of $SubIt$ iterations) using a local search algorithm as explained in Section 5.4.4. Within this loop, some small changes are made to the best-known solution by removing the jobs assigned to the sequences of one or two machines randomly selected in order to reassign them to new machines and create a new sequence of jobs. Finally, the proposed ALNS algorithm terminates when the maximum number of iterations ($MaxIt$) is reached in the main loop of the algorithm.

A brief review of the proposed ALNS metaheuristic algorithm is shown in the pseudo-code given in Figure 5.3. In this pseudo-code, the index *it* refers to the iteration of the main algorithm while *it1* represents the sub-loop iteration of the local search algorithm. Generally, the full algorithm has the following steps:

- **Step 0** (lines 1 to 7 of the pseudo-code): Set the weights of all removal and construction heuristics to 1 and create a feasible solution (Sol^0) using a construction heuristic following the procedure described in Section 5.4.2. Note that at this point (Sol^0) is known as the current best solution (Sol^*).
- **Step 1** (lines 9 to 10 of the pseudo-code): Define the probabilities of each removal and construction heuristics and select a removal-construction pair using the roulette wheel selection as described in Section 5.4.5.
- **Step 2** (line 11 of the pseudo-code): Apply the selected removal and construction heuristics on the solution (Sol^{it}) to generate a new solution (New_Sol^{it}).
- **Step 3** (lines 12 to 23 of the pseudo-code): Apply the decision rule to accept or reject New_Sol^{it} using the current temperature and the damping ratio. Update the current best solution ever found so far (Sol^*) according to the procedure described in Section 5.4.6.
- **Step 4** (line 24 of the pseudo-code): Give a score to the heuristics employed.
- **Step 5** (lines 25 to 29 in the pseudo-code): Apply the local search algorithm to improve the current best solution ever found so far (Sol^*).
- **Step 6** (line 30 in the pseudo-code): Update the weights of removal and construction.
- **Step 7** (lines 31 to 32 of the pseudo-code): If the algorithm is terminated, output the current best solution ever found (Sol^*). Otherwise, go to **Step 1**.

Set the maximum number of iterations (*MaxIt*) and sub-iterations (*SubIt*).
Run C1, C2, C3 and C4 heuristics to define an initial solution (Sol^0).
Define the initial weights $rw_r^- = 1$ where $r \in R$ and $iw_c^+ = 1$ where $c \in C$;
$Sol^* = Sol^0$;
$it = 0$;
$T = Tem$;
$\alpha = redu$;
While $it < MaxIt$
  Run Eqs. (5.31) and (5.32) to compute P_r^{rw} and P_c^{iw};
  Perform the roulette wheel selection using removal (P_r^{rw}) and construction (P_c^{iw}) probabilities.
  Apply these selected removal and construction heuristics on solution (Sol^{it}) to build a new solution New_Sol^{It}.
  If $Z(New\_Sol^{it}) \leq Z(Sol^*)$
    $Sol^* = New\_Sol^{it}$;
    $Sol^{it+1} = Sol^*$;
  Elseif $Z(New\_Sol^{it}) > Z(Sol^*)$
    $\Delta = \left|Z(New\_Sol^{it}) - Z(Sol^*)\right|$
    $p = e^{-\Delta/T}$
    If $rand_1 \leq p$
      $Sol^{it+1} = New\_Sol^{it}$;
    Else
      $Sol^{it+1} = Sol^{it}$;
    Endif
  Endif
  Give a score to the selected removal and construction heuristics using Eq. (5.30).
  $it1 = 0$;
  While $it1 < SubIt$
    Apply the local search algorithm to improve the current best solution found so far (Sol^*).
    $it1 = it1 + 1$;
  Endwhile
  $it = it + 1$;
  Update the heuristics weights (rw_r^- and iw_c^+) using Eqs. (5.28) and (5.29).
Endwhile
Output the best solution ever found so far (Sol^*).

Figure 5.3 Pseudo-code of the proposed ALNS metaheuristic algorithm

5.5 Computational results

In order to illustrate the application of our algorithm, we first define several test problems, each of them having a different level of complexity. Then, using these tests, the performance of our ALNS algorithm is evaluated by comparing its solutions to the exact solutions obtained using CPLEX[8] solver and to the solutions obtained by two well-used metaheuristic algorithms in the literature, namely, SA (Van Laarhoven, & Aarts, 1987) and VNS (Mladenović, &

[8] https://www.ibm.com/analytics/cplex-optimizer

Hansen, 1997). The parameters of all algorithms have been calibrated to improve their performance and to perform an unbiased analysis. Finally, sensitivity analyses on the proposed ALNS metaheuristic and our scenario-based robust optimization model are performed. Except for the exact solution obtained using GAMS software, all metaheuristic algorithms are implemented in MATLAB[9] software and computed on a laptop with an Intel(R) Core (TM) i7-10850H CPU @ 2.70GHz 2.71 GHz.

5.5.1 Data generation

Since our SDPFSP has special attributes and parameters that have not been defined in any relevant distributed permutation flow-shop model, existing benchmarks in the literature (Ruiz et al., 2019) are not suitable for the proposed optimization problem. Therefore, based on the logic reported in Fathollahi-Fard et al., (2021) and using relevant models in the area of SDPFSP (Fathollahi-Fard et al., 2021; Lu et al., 2021), we generated our own test instances.

In this regard, 12 test problems are generated according to three levels of complexity. Tests T1 to T4 are small size problems, tests T5 to T8 are medium size problems and tests T9 to T12 are large size problems (Table 5.1). The range of values that each parameter used in these tests can take is provided in Table 5.2. It should be noted that the robustness factor (λ) used in Eq. (5-5) is set at 0.5 (Fathollahi-Fard, et al., 2022) in all tests.

[9] https://www.mathworks.com/products/matlab.html

Table 5.1 Definition of instances

Level of complexity	Instances	Size of instances				
		Number of factories (F)	Number of machines (M)	Number of operating modes (P)	Number of jobs (N)	Number of scenarios (S)
Small	T1	2	2	2	4	3
	T2	2	2	2	8	3
	T3	2	4	2	10	3
	T4	3	4	2	12	3
Medium	T5	3	6	2	16	6
	T6	3	6	3	20	6
	T7	4	6	3	24	6
	T8	4	6	3	30	6
Large	T9	6	8	3	40	9
	T10	6	8	4	46	9
	T11	6	8	4	50	9
	T12	6	8	4	60	9

Table 5.2 Range of parameters of our optimization model

Parameter	Range
NPC_{nmpfs}	*randi*([2, 8],N, M, P, F, S)
CO_{mpf}	*randi*([8, 20],M, P, F)*10^4
JO_{mpf}	*randi*([2, 9],M, P, F)
CJ_{mpf}	*randi*([8, 20],M, P, F)
LD_{mpf}	*randi*([8, 30],M, P, F)
LBJ	*round* (*sum*($JO_{mpf}/3$))
UBL	$round\left(sum\left(LD_{mpf} * \left(\frac{2}{3}\right)\right)\right)$
$time_s$	*randi*([1, 16],1,S)
RW_{mpf}	*rand*(M, P, F)*0.1
IEC_{mf}	(*randi*([1, 10],M, F)+*rand*())*10^5
UEC_{mpf}	(*randi*([10, 40],M, P, F)+*rand*())*10^5
EC_{mpf}	(*randi*([1, 10],M, P, F)+*rand*())*10^5
$UBEC$	$round\ (sum((IEC_{mpf} + UEC_{mpf} + EC_{mpf}) * (\frac{2}{3})))$
B	*randi*([*round*(*sum*(JO_{mtf}.* $CJ_{mtf} + CO_{mtf}$)/2), *round*(*sum*(JO_{mtf}.* $CJ_{mtf} + CO_{mtf}$))
MS_{mpfs}	*rondi*([0,1], M, P, F, S)*0.8)
H_{nimpfs}	*round(rand*(N, N, M, P, F, S)*0.9)
RP_{mpfs}, AV_{mpfs}	**if** $MS_{mpfs} == 0$ $RP_{mpfs} = Unif\ (sum(H_{nimpfs} * PC_{nmpfs}), 2 * sum(H_{nimpfs} * PC_{nmpfs})$**else** $AV_{mpfs} = 0$ **end**
γ_{mp}	$\dfrac{1}{7 * sum(H_{nimpfs} * NPC_{nmpfs})/S}$
δ_{mp}	$\dfrac{1}{\frac{3}{2} * sum(H_{nimpft} * NPC_{nmpf})/S}$
MW	**if** *sum*(RW_{mpf})>1 *randi*([*round*(*sum*(RW_{mpf})/2), *round*(*sum*(RW_{mpf}))**else** *rand*()+(*sum*(RW_{mpf})/2)**end**

**randi* creates random integer numbers between a lower and an upper bound.
**rand* creates random continuous numbers between zero and one.
**Unif* creates random continuous numbers between a lower and an upper bound.
**round* transforms continuous numbers to the closest integer numbers.
**sum* sums numbers in a matrix.
**Exp* is the exponential distribution function.

5.5.2 Calibration of parameters for our ALNS

In any solving process using metaheuristics, the adjustment of the parameters is of crucial importance (Pasha et al., 2022). Most of the time, a well-tuned metaheuristic solves optimization problems with the highest level of efficiency (Liu et al., 2017b). Generally, the

ALNS is very sensitive to the scores attributed according to the recorded performances. Thus, an adequate calibration is necessary before being able to compare its global performance with that of an exact solver or other metaheuristics (Mara et al., 2022). This adjustment requires a judicious choice of values for each of the following input parameters: *MaxIt, SubIt, redu, Q,* ϖ_1, and ϖ_2. Note that $\varpi_3 = 1 - \varpi_1 - \varpi_2$.

For each of the six independent parameters, we consider three candidate values listed in Table 5.3. If we were using a full factorial method, we would have to run a total of 3^7=2187 experiments for each test problem which did not make sense. In order to reduce the number of experiments to be performed, this study applies Taguchi' experimental design method (Karna, & Sahai, 2012). This method uses a set of predefined orthogonal arrays to reduce the number of experiments to be performed. The orthogonal array properties are such that, between each pair of columns, each combination of candidate values appears an equal number of times.

Table 5.3 Candidate values of the parameters for the proposed ALNS metaheuristic

Parameter	Levels of candidate values		
	1	2	3
$MaxIt$	500	1000	2000
$SubIt$	20	30	50
$redu$	0.9	0.99	0.999
Tem	10000	15000	20000
Q	0.1	0.3	0.5
ϖ_1	0.2	0.4	0.6
ϖ_2	0.1	0.3	0.4

Since we have three candidate values and seven parameters, the Taguchi method suggests the use of an orthogonal array having 27 lines and seven columns (L_{27}). Thus, with this structure, only 27 experiments are needed for each test problem in order to find the best set of values for the input parameters.

The selection of the parameters values is done using the relative percentage difference (RPD). For a minimization problem like the one defined in our model, the RPD is calculated as follows (Fard, & Hajaghaei-Keshteli, 2018):

$$RPD = \frac{Alg_{sol} - Min_{sol}}{Min_{sol}} \tag{5.36}$$

where Alg_{sol} represents the mean of the objective function values obtained for all the tests at each row of the orthogonal array. As such, Min_{sol} is the minimum value of these mean values among the 27 experiments. Based on the results shown in Table 5.4, the set of candidate values corresponding to the smallest mean value of RPD are selected.

Table 5.4 Orthogonal array and relative percentage of deviation of ALNS metaheuristic's parameters tuning

Number of experiments	Levels of candidate values for each parameter							Mean values from all tests	RPD
	$MaxIt$	$SubIt$	$redu$	Tem	Q	ϖ_1	ϖ_2		
L1	1	1	1	1	1	1	1	1206.234	0.024834
L2	1	1	1	1	2	2	2	1210.047	0.028074
L3	1	1	1	1	3	3	3	1217.597	0.034489
L4	1	2	2	2	1	1	1	1177.004	0
L5	1	2	2	2	2	2	2	1212.439	0.030106
L6	1	2	2	2	3	3	3	1203.122	0.02219
L7	1	3	3	3	1	1	1	1213.383	0.018842
L8	1	3	3	3	2	2	2	1200.015	0.019551
L9	1	3	3	3	3	3	3	1215.035	0.032311
L10	2	1	2	3	1	2	3	1224.644	0.040476
L11	2	1	2	3	2	3	1	1218.26	0.035052
L12	2	1	2	3	3	1	2	1213.547	0.031048
L13	2	2	3	1	1	2	3	1207.632	0.026023
L14	2	2	3	1	2	3	1	1198.415	0.018191
L15	2	2	3	1	3	1	2	1228.571	0.043812
L16	2	3	1	2	1	2	3	1208.206	0.02651
L17	2	3	1	2	2	3	1	1196.637	0.016681
L18	2	3	1	2	3	1	2	1203.843	0.022803
L19	3	1	3	2	1	3	2	1211.279	0.029121
L20	3	1	3	2	2	1	3	1236.685	0.050706
L21	3	1	3	2	3	2	1	1229.511	0.044611
L22	3	2	1	3	1	3	2	1223.275	0.039313
L23	3	2	1	3	2	1	3	1194.813	0.015131
L24	3	2	1	3	3	2	1	1202.008	0.021244
L25	3	3	2	1	1	3	2	1211.929	0.029673
L26	3	3	2	1	2	1	3	1209.606	0.027699
L27	3	3	2	1	3	2	1	1206.745	0.025268

The results of our analyses are shown in Figure 5.4. Note that we compute the average value of RPD for each factor in each value. Although the lowest value of RPD is for the experiment L4 from Table 5.4, we used the mean RPD to consider all the experiments for finding the calibrated values. According to these results, the tuned value of each parameter is reported in Table 5.5.

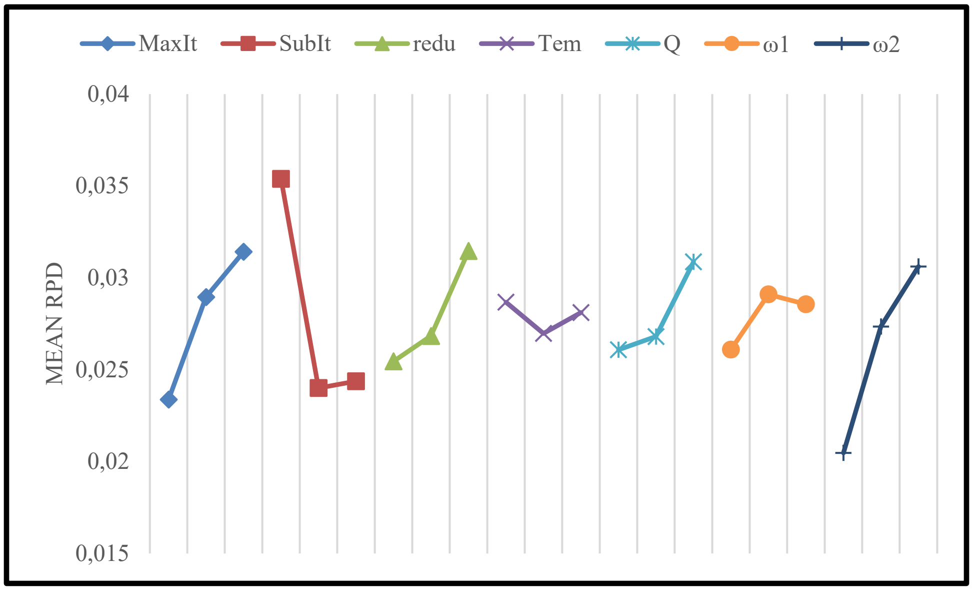


Figure 5.4 Average value of RPD computed for each level of candidate values and for each parameter of the ALNS metaheuristic algorithm

Table 5.5 Calibrated values of our ALNS metaheuristic algorithm

Parameter	Selected value
$MaxIt$	500
$SubIt$	30
$redu$	0.9
Tem	15000
Q	0.1
ϖ_1	0.2
ϖ_2	0.1
ϖ_3	0.7

5.5.3 Comparison of our ALNS with the exact solver and other metaheuristic algorithms

The proposed ALNS metaheuristic is validated by comparing its result to those of the exact solver and two popular and well-used metaheuristics, namely SA and VNS. The selection of these algorithms for comparison purposes is based on their similarity to ALNS. Hence, these three algorithms are neighborhood-based metaheuristics. The four SA parameters $MaxIt$, *SubIt,* $redu$ and Tem were set to 500, 30, 0.9 and 15000 respectively. For the VNS, the $MaxIt$ and $SubIt$ parameters were set to the same values while the number of neighborhood procedures *(K)* was fixed to four. To make this comparison, we used the neighborhood procedures developed in Fathollahi-Fard et al., (2021). For all instances, we run each metaheuristic 10 times. Of all the results obtained, the best (B) and the worst (W) results were identified and the mean (M) and standard deviation (STD) were calculated. Finally, the average CPU time and the optimality gap (OG) between the mean value and the exact solution were also evaluated when possible. Indeed, due to the high complexity of large-scale instances, we were facing the problem of running out of memory when attempting to solve them using the exact solver. Consequently, the termination criterion for the exact solver was set to 3600 seconds to prevent this problem from occurring, and optimal solutions could not be identified for tests T8 to T12. For the metaheuristic algorithms, we also considered a maximum time of 1000 seconds to identify a solution each time the maximum number of iterations was not reached. Thus, the metaheuristics were forced to identify a solution in a reasonable time.

Table 5.6 Results obtained using our ALNS metaheuristic algorithm, the exact solver, and, VNS and SA algorithms

Algorithms		T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
Exact solver	Solution	49.3622	88.3642	242.4836	310.2739	528.1732	742.1822	895.7218	-	-	-	-	-
	CPU (s)	15.7362	37.2736	215.237	1037.2832	2484.5783	3600	3600	-	-	-	-	-
ALNS	B	54.2222	99.1111	256.8889	325.7778	571.8611	828.8475	1084.273	1376.347	2457.303	3319.621	3583.22	4285.669
	W	54.2222	99.1111	256.8889	327.5556	588.5833	932.3611	1169	1411.667	2586.76	3499.311	3686.347	4480.444
	M	54.2222	99.1111	256.8889	326.2667	576.8	883.2339	1099.497	1386.747	2530.773	3426.408	3624.262	4394.503
	STD	0	1.5E-14	0	0.708957	7.0115	46.2319	26.6256	11.10219	40.4757	62.19828	35.4412	61.0313
	Average CPU (s)	51.7759	141.2894	504.9767	1000	1000	1000	1000	1000	1000	1000	1000	1000
	Average OG (%)	9.84	12.16	5.94	5.15	9.2	19	22.74	-	-	-	-	-
VNS	B	54.2222	99.1111	256.8889	325.7778	571.8611	830.8611	1106.778	1390.083	2504.198	3322.866	3611.176	4309.293
	W	54.6667	103.5556	261.7778	333.3333	590.9167	946.75	1169	1429.361	2696.951	3522.702	3777.162	4498.045
	M	54.2666	99.5555	258.2222	328.8444	580.5722	903.5274	1124.783	1413.942	2601.067	3424.548	3691.916	4393.999
	STD	0.1405	1.4054	2.042092	2.603781	6.8733	47.5324	17.12539	17.17436	81.81583	66.73928	51.75795	55.21337
	Average CPU (s)	1.0829	3.9082	7.6411	13.99444	46.2062	65.90001	102.7823	259.1573	1000	1000	1000	1000
	Average OG (%)	9.93	12.66	6.49	5.9852	9.92	21.73	25.57	-	-	-	-	-
SA	B	54.2222	99.1111	257.3333	330.2222	579.8333	845.6632	1106.778	1390.083	2504.657	3337.391	3607.01	4503.244
	W	54.6667	99.1111	263.5556	334.2222	637	946.75	1169	1445.111	2707.284	3593.558	3755.446	4699.877
	M	54.2666	99.1111	259.5111	332.0889	599.8222	903.6763	1129.217	1422.556	2620.628	3467.05	3668.261	4590.257
	STD	0.14056	1.5E-14	2.251198	1.673487	25.0477	43.62635	22.23218	15.80339	75.42499	104.107	50.7244	64.66478
	Average CPU (s)	1.0484	0.92796	2.05106	6.85824	11.47991	16.7465	25.4147	62.81759	389.3883	1000	1000	1000
	Average OG (%)	9.93	12.16	7.02	7.03	13.56	21.75	26.06	-	-	-	-	-

The performance of our ALNS metaheuristic algorithm against the exact solver as well as the SA and VNS approaches can be compared using the results presented in Table 5.6 where the solution represents the optimal solution found by the exact solver. For each test, the problem was solved 10 times using each of the metaheuristics (ALNS, VNS, SA).

Based on the results of Table 5.6, we can say that the solutions of the metaheuristics are very close to the optimal solution identified using the exact solver especially in small instances. For all 12 instances, the best solution among the 10 solutions identified by each of the 3 algorithms, was the lowest one for the ALNS. However, the main difference can be seen in the range of average solutions and standard deviation for each metaheuristic. In addition, ALNS show lower OG values than other two metaheuristics indicating a lower deviation of its solutions from the exact solution. The optimality gaps are calculated by the deviation of the solution of the exact solver with the average of the solutions of our metaheuristic algorithms, which can be seen in Figure 5.5. As mentioned before, for instances T8 to T12, the exact solver cannot find a solution and consequently, the respective OG values could not be computed. Instead, the average improvement done by the metaheuristics for these instances, is shown in Figure 5.6. This improvement is based on the percentage of deviation of the initial solution with the best solution ever found. Finally, the accuracy of the algorithms is analyzed statistically in Figure 5.7 with emphasis on the standard deviations of all solutions.

Figure 5.5 shows that in general, ALNS is stronger than SA and VNS in finding an optimal solution as is evident in T3 to T7 where ALNS finds better solutions compared to VNS and SA. As such, VNS is better than SA in a few instances including T3 to T5. Generally, the behavior of algorithms based on the criterion of the average OG is very competitive and all the algorithms are reliable on the basis of the optimality gap from the best solution ever found especially in small instances from T1 to T6 where the OG is less than 0.2.

Figure 5.6 shows the average improvement brought by the metaheuristics based on the comparison of the initial solutions with the best solutions. In this regard, we compare the final solutions obtained by the algorithms with the best initial solution from our constructive heuristics. For all complex instances, i.e., from T8 to T12, the highest relative improvement is achieved by ALNS and this algorithm shows better performance in comparison with SA and VNS. For example, in test instance T9, the ALNS shows an improvement of approximately more than 20 percent, while SA and VNS improved the initial solution by approximately less than 20 percent.

The accuracy of the metaheuristics was analyzed using the normalized standard deviation of the results. To this end, the normalized values of each instance for each algorithm are computed and based on their range, a box plot is shown in Figure 5.7. A lower value for these boxes is preferable when these results confirm that the local optimal solution found by ALNS is more accurate than those of SA and VNS.

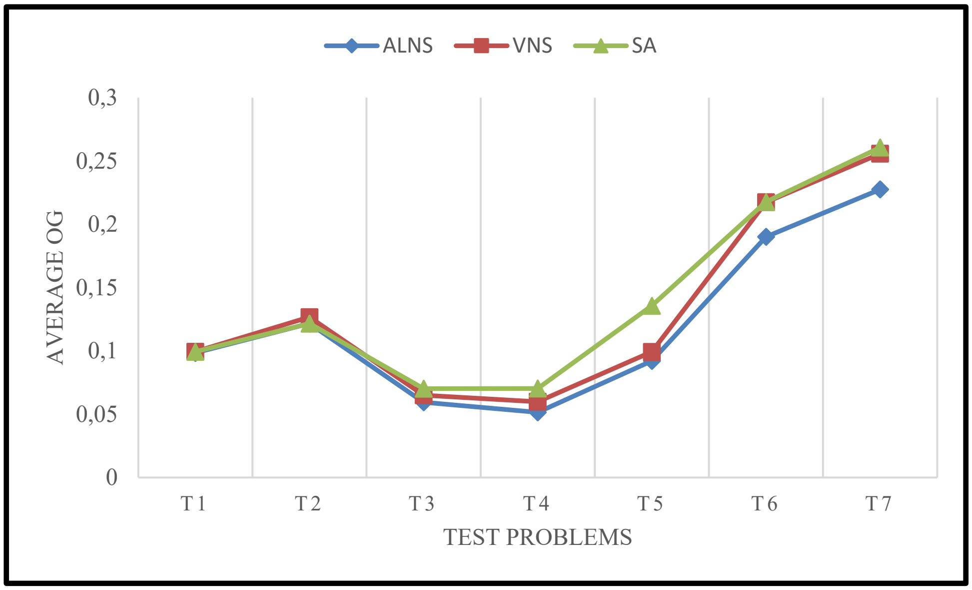


Figure 5.5 Average optimality gaps of the metaheuristics.

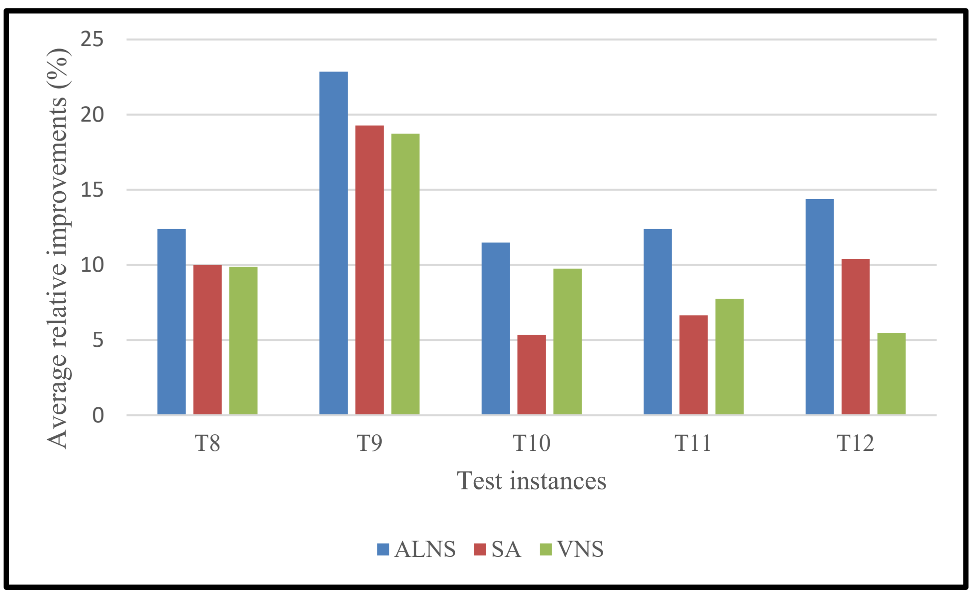


Figure 5.6 Average improvement done by the metaheuristics

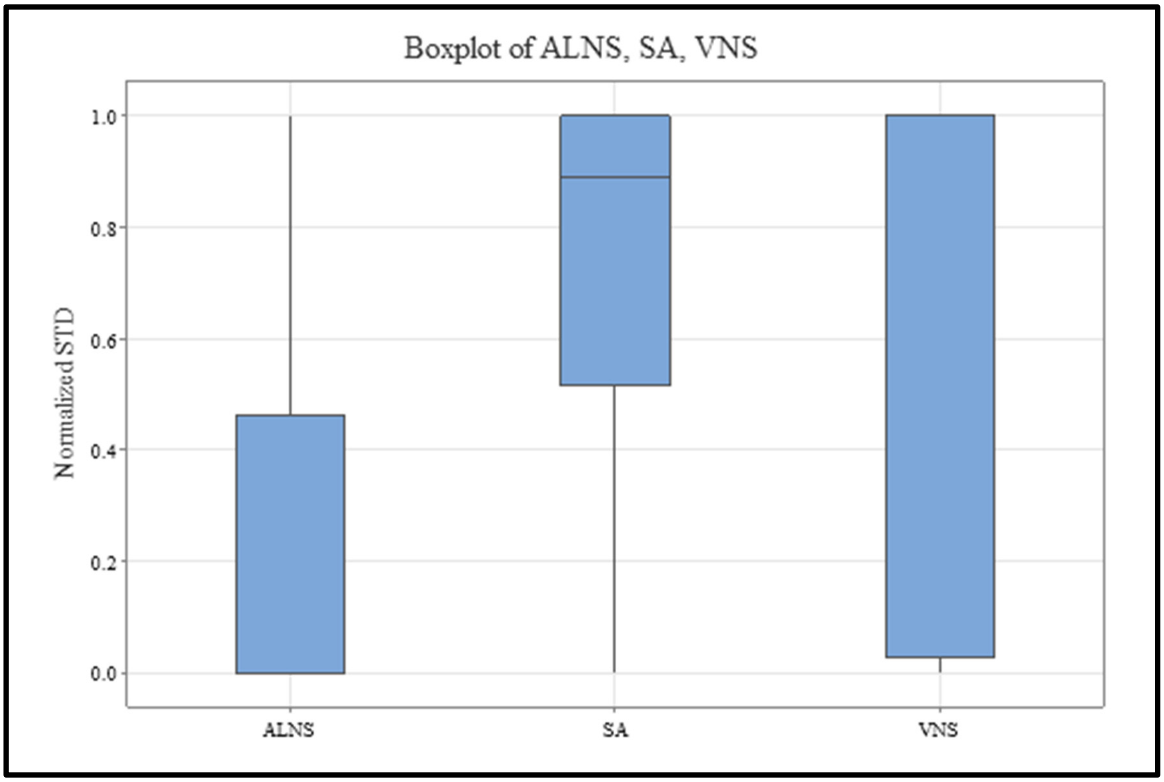


Figure 5.7 Normalized standard deviations of our metaheuristics based on the box plot

5.5.4 Sensitivity analyses of the performance of the removal-construction heuristics

A significant contribution of our ALNS metaheuristic is the use of different removal and construction heuristics to effectively destroy and repair solutions. An open question is which of these heuristics is the most effective and has the key role of helping the main algorithm to find an optimal solution? In order to answer this question, we will here analyze each pair of removal-construction heuristics according to their effectiveness. It should be noted that the comparison of removal and construction heuristics individually has been done several times in the literature of ALNS (Mara et al., 2022).

Here, we select a test problem like T4 to perform these analyses. To perform our sensitivity analyses, we redesign the tuned metaheuristic algorithm to use only one pair of removal-construction heuristics in all iterations. Since there are six removal and four construction heuristics, there are a total of 24 = 6× 4 pairs of heuristics in our sensitivity analysis. We have considered the maximum of 100 seconds for each version of ALNS in these analyses. It should be noted that each version of ALNS has been run 10 times and the average relative improvements are reported. Table 5.7 presents the results of our analysis where the main criterion is the relative improvement of the initial solution made by the metaheuristic. In this regard, the RPD metric described in Eq. (5.36) is used where Min_{sol} and Alg_{sol} are respectively replaced by the initial ($Z(Sol^{it})$) and the best-known ($Z(Sol^{*})$) solutions found by the ALNS metaheuristic. Moreover, Figure 5.8 and Figure 5.9 respectively show the average relative improvement that brings each of the removal and construction heuristics individually.

Table 5.7 Relative improvement of each pair of removal-construction heuristics (the best and worst results are shown in bold)

Removal heuristics	**Construction heuristics**	**Average improvement induced by the pair of removal-construction heuristic (%)**
R1	C1	8.34752
	C2	11.62613
	C3	11.6652
	C4	11.51931
R2	C1	**7.94358**
	C2	11.61427
	C3	11.62144
	C4	11.54663
R3	C1	10.48756
	C2	11.2349
	C3	11.40277
	C4	11.65785
R4	C1	**11.71702**
	C2	11.49871
	C3	11.3008
	C4	11.54885
R5	C1	9.05299
	C2	9.04783
	C3	11.45021
	C4	11.44587
R6	C1	8.69896
	C2	11.46428
	C3	11.53808
	C4	11.28246

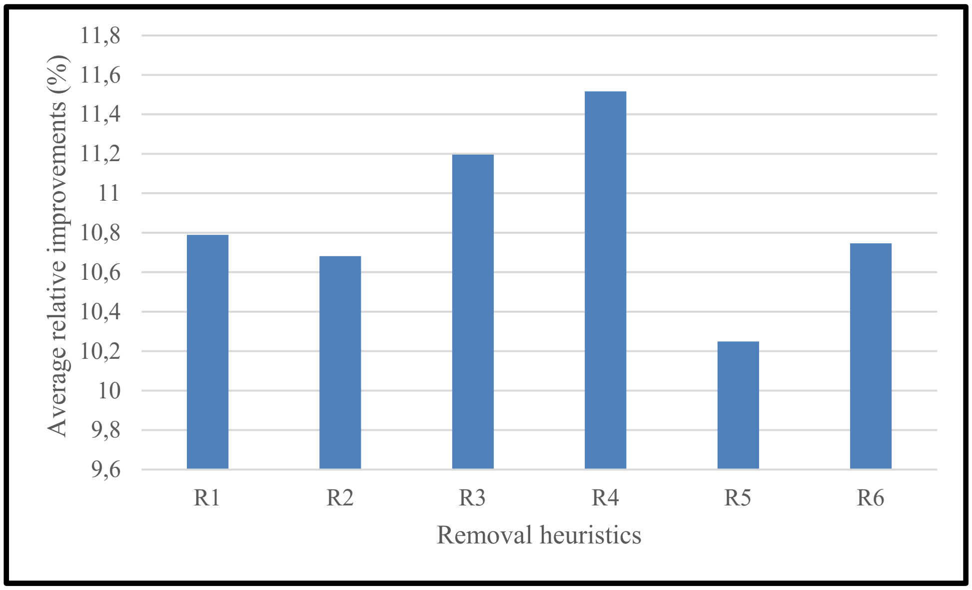


Figure 5.8 Average relative improvement induced by each of the removal heuristics

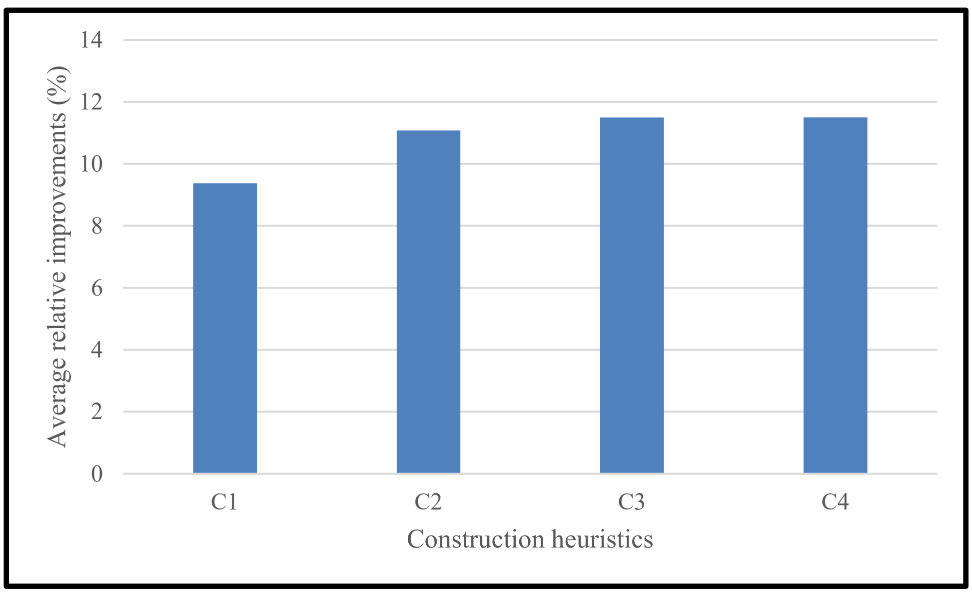


Figure 5.9 Average relative improvement induced by each of the construction heuristics

According to the results provided in Table 5.7, the pair of heuristics composed of R4 (i.e. the removal heuristic based on the makespan of the machines) and C1 (i.e. the combination of decision rules NR1 and AF1) is the most effective for exploring new solutions in the search space. Conversely, the pair formed by R2 (which is based on the random-operation-mode-based removal) and C1 turns out to be the least effective among all the pairs of heuristics. These findings highlight the role of removal heuristics on the effectiveness of construction ones.

As indicated in Figure 5.8, among the removal heuristics, the fourth (R4) proves to be the most efficient while the fifth (R5) is very inefficient. According to results of Figure 5.9, it is the first constructive heuristic (C4, which combines the decision rules NR2 and AF2), which is the best method to reconstruct a solution. Conversely, the first constructive heuristic (C1) has the lowest efficiency compared to other construction heuristics.

5.5.5 Sensitivity analyses of the scenario-based robust optimization model

In order to prove the applicability of our optimization model, we first introduce our industrial example. Then, sensitivity analyses are performed on the robust optimization coefficient to show the impact of robust programming on the results compared to traditional stochastic programming. Finally, the main parameters of our optimization model are also analyzed to highlight their impact on the value of the objective function. All these analyses were performed by the exact solver to find an optimal solution.

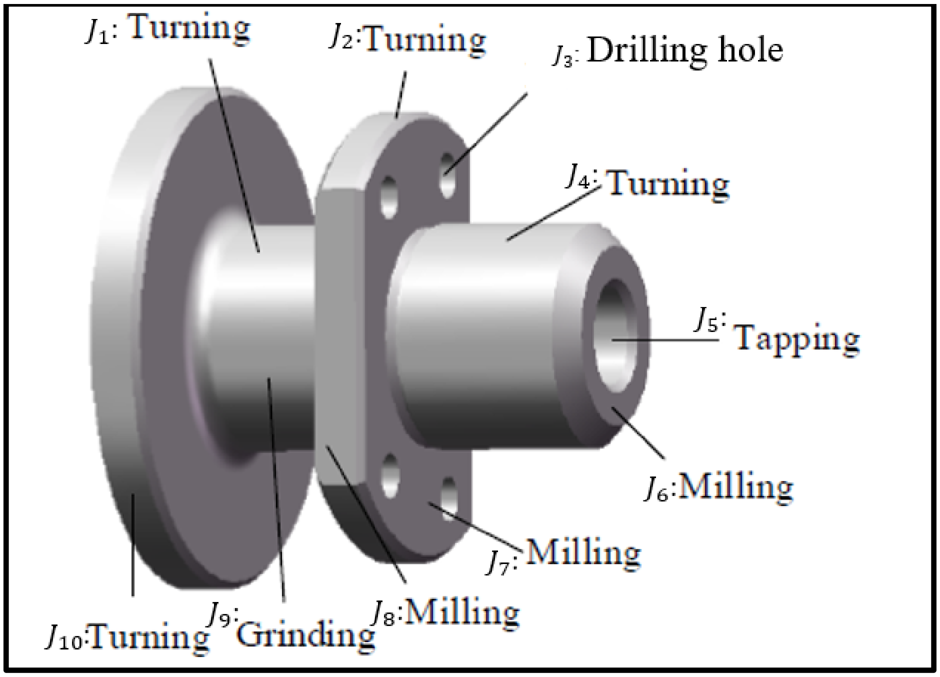


Figure 5.10 Flange considered for production in our industrial example

The industrial example considered consists of the production of a flange like the one shown in Figure 5.10. Ten jobs identified as J_1 to J_{10} are performed by five different CNC machines located in one factory to produce the flange. Three probabilistic scenarios are considered in this case study to represent all pessimistic (S_1), realistic (S_2) and optimistic (S_3) cases. For each machine, two operating modes for the manual and automatic versions of our CNC machines are considered. The job processing time is shown in Table 8 while the parameters related to energy consumption (UEC_{mpf}, EC_{mpf}) and sustainability (CO_{mpf}, , CJ_{mpf}, JO_{mpf}, LD_{mpf}, RW_{mpf}) are provided in Table 5.9.

Table 5.8 Processing time of jobs in minutes

Machines	Scenarios	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
Manual mode	S_1	6	6	7	6.8	6	6	6	5.5	6.4	7
	S_2	5.9	5.8	6	6.5	5	5.5	5.7	5.3	5.8	6.7
	S_3	5.7	5.7	5.5	6.4	4	5	5.5	5	5	6.5
Automatic mode	S_1	5.6	5	4.8	5.3	4	3.7	3.4	3.6	4.3	5.6
	S_2	5.45	4.9	4.7	5.2	3.9	3.6	3.3	3.5	4.2	5.5
	S_3	5.35	4.8	4.6	5.1	3.8	3.5	3.2	3.4	4.1	5.4

Table 5.9 Values of the parameters related to the energy consumption and sustainability used in the industrial example

Machines	UEC_{mpf} (KWh)	EC_{mpf} (KWh)	CO_{mpf} (\$)	CJ_{mpf}(\$)	$JO_{mpf}(Person)$	$LD_{mpf}(Days)$	RW_{mpf}
Automatic mode	4.1	2.9	32.4×10^3	2	3	7	0.04
Manual mode	4.3	3.2	20.4×10^3	1	8	2	0.14

To demonstrate how this solution from our case study can enhance sustainability criteria, we have calculated the energy consumption, the workforce employed, and the count of lost workdays as follows:

$$UBEC_{sol} = \sum_{m\in M}\sum_{p\in P}\sum_{f\in F}(Y_{mpf}\times EC_{mpf}) \tag{5.37}$$

$$+\sum_{m\in M}\sum_{p\in P}\sum_{f\in F}\sum_{n\in N}\sum_{s\in S}\pi_s\left(UEC_{mpf}\times Y_{mpf}\times PC_{nmpfs}\right)$$

$$+\sum_{m\in M}\sum_{f\in F}IEC_{mf}\times\sum_{p\in P}Y_{mpf}$$

$$LBJ_{sol} = \sum_{m\in M}\sum_{p\in P}\sum_{f\in F}(Y_{mpf}\times JO_{mpf}) \tag{5.38}$$

$$UBL_{sol} = \sum_{m\in M}\sum_{p\in P}\sum_{f\in F}(Y_{mpf}\times LD_{mpf}) \tag{5.39}$$

where $UBEC_{sol}$, LBJ_{sol}, and UBL_{sol} are respectively the amount of energy consumption, the workforce employed, and the count of lost workdays for the solution obtained from our case study. Then, we can compute the overall improvement to these ideal bounds of sustainability criteria as follows:

$$RPD_{UBEC} = \frac{UBEC - UBEC_{sol}}{UBEC} \quad (5.40)$$

$$RPD_{LBJ} = \frac{LBJ_{sol} - LBJ}{LBJ} \quad (5.41)$$

$$RPD_{UBL} = \frac{UBL - UBL_{sol}}{UBL} \quad (5.42)$$

where RPD_{UBEC}, RPD_{LBJ}, and RPD_{UBL} are respectively the amount of improvement to energy consumption, job opportunities, and lost workdays. Following the computation of these values from our solution, we can conclude that our solution has the potential to enhance energy consumption by 24%, increase job opportunities by 67%, and reduce lost workdays by 18%, as compared to the ideal bounds of these parameters.

With a robust coefficient (λ) adjusted to 0.5, an objective function value of about 35.2 minutes was identified in a computation time of 11.54 seconds using the exact solver. A value of $\lambda = 0$ corresponds to the application of traditional stochastic programming. Here, three values were considered: 0, 0.5 and 1. The values obtained for the objective function are shown in Figure 5.11. These results confirm that an increase in the robust coefficient increases the value of the objective function.

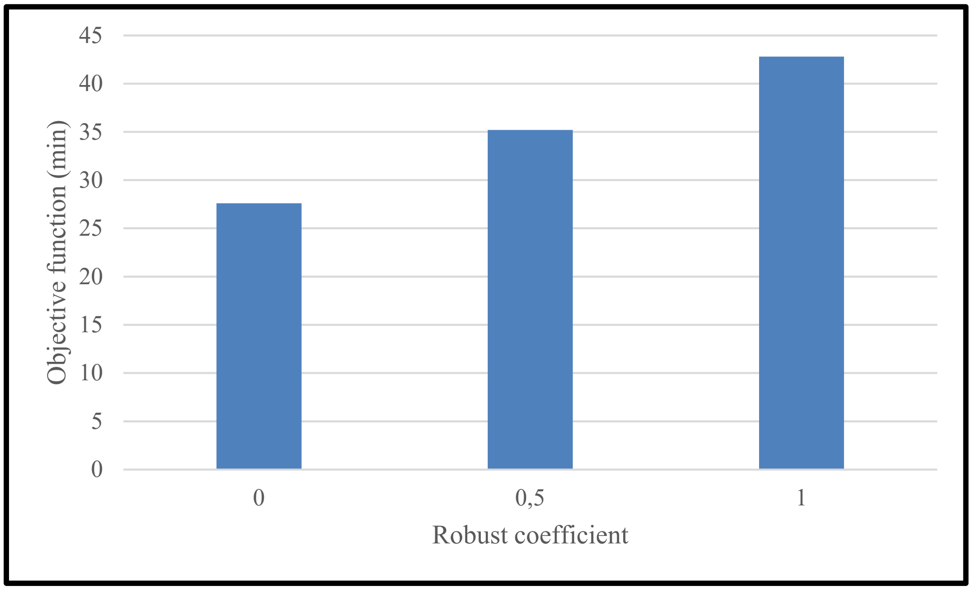
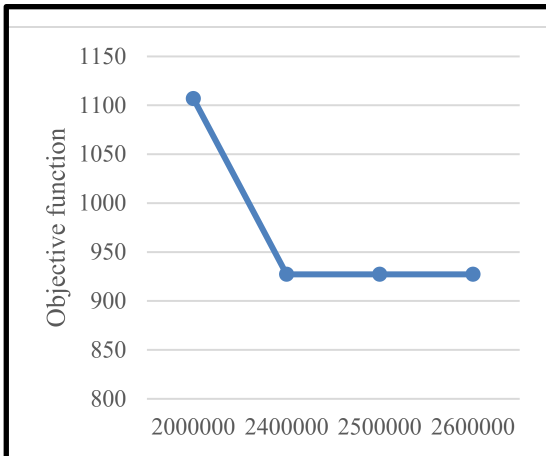


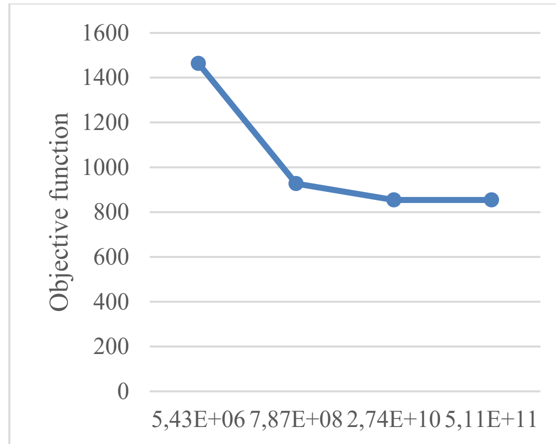
Figure 5.11 Sensitivity analysis on the robust coefficient

To analyze the sustainability criteria, we focused on the financial budget (B) as the main economic criterion, the maximum authorized amount of energy consumed by machines ($UBEC$) as an environmental factor, and the minimum authorized number of workers employed (LBJ) as well as the maximum authorized number of lost workdays (UBL) as social factors. To perform these analyses, the instance T4 is selected randomly. Moreover, only one probabilistic scenario is considered here to ignore the impact of uncertainty and focus on sustainability criteria. In this regard, for each parameter, four cases are considered and the value of the objective function is analyzed for each of these case as shown in Figure 5.12.

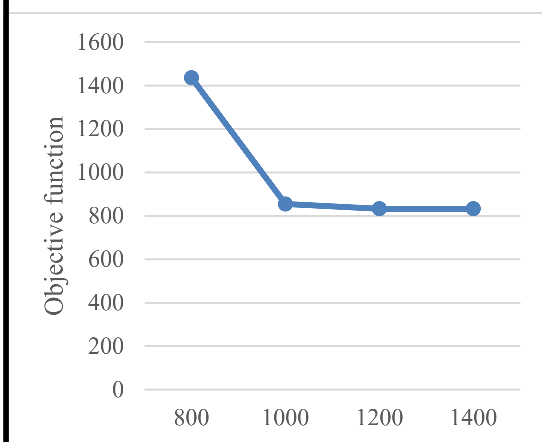
Looking at these graphs, the main finding is that an increase in the budget, the maximum allowed amount of energy consumed by the machines, and the maximum allowed number of lost workdays can improve the quality of our solution. However, an increase in the minimum number of employed workers allowed degrades the quality of the solution.



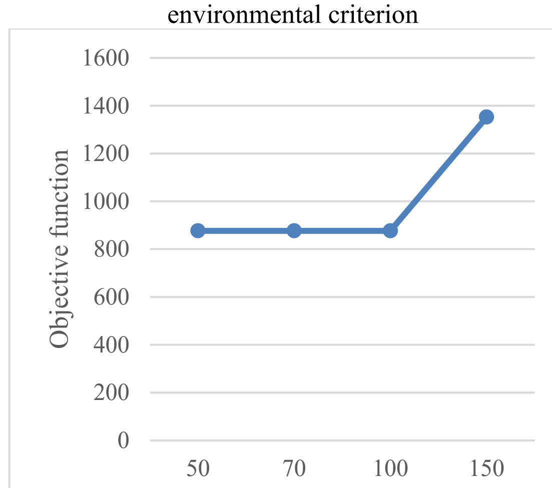
(a) Variation of the objective function according to the budget as economic criterion



(b) Variation of the objective function according to the maximum amount of energy allowed as environmental criterion



(c) Variation of the objective function according to the maximum number of lost workdays allowed as social criterion



(d) Variation of the objective function according to the minimum number of workers employed allowed as social criterion

Figure 5.12 Sensitivity analyses with regards to the sustainability criteria

5.6 Discussions, and managerial insights

In this section, we engage in a comprehensive discussion of our research, encompassing our contributions, results, and the managerial insights gleaned from our study, all within the context of our case study. Our research centers around addressing the SDPFSP, a multifaceted optimization challenge that takes into account economic, environmental, and social criteria in the realm of production scheduling. An important facet of our approach is its consideration of various uncertainties, including machine breakdowns, processing time variations, and unpredictable arrivals of new jobs. The primary contribution of our research lies in the development of a scenario-based robust optimization model, designed to minimize the expected makespan while managing deviations within probabilistic scenarios.

To tackle the intricacies of the SDPFSP, we introduced a new ALNS metaheuristic algorithm. ALNS employs four constructive heuristics to establish an initial solution, followed by a judicious process of solution refinement through efficient removal and construction heuristics, complemented by a local search algorithm to explore the solution space further. The comparative analysis highlights that our metaheuristic solutions closely approach the optimal solutions obtained through the exact solver, particularly in smaller instances. ALNS consistently outperforms other metaheuristics, as evidenced by lower optimality gap values, indicating solutions that are closer to the optimum. Particularly noteworthy is ALNS's capacity to significantly improve upon initial solutions in complex instances, underscoring its robustness.

To fine-tune ALNS for optimal efficiency, we leveraged calibration techniques, notably the RPD metric, resulting in notable improvements in algorithm performance, as indicated in Table 5.5. Furthermore, the calibration was supported by a lower average optimality gap, as depicted in Figure. 5.5, and substantial relative improvements when comparing initial solutions to the best solutions, illustrated in Figure 5.6. Statistical analyses, represented in Figure 7, reinforced ALNS's prowess. Notably, we dissected and assessed individual heuristics within ALNS, identifying R4 as the most effective removal heuristic and C1 as the top-performing construction heuristic.

Our research extended beyond algorithmic development, demonstrating the practical applicability of SDPFSP through an industrial numerical example, i.e., the production of a flange using CNC machines. In doing so, we validated the impact of our robust programming model on solution quality in comparison to stochastic programming, as illustrated in Figure 5.11. Based on the results derived from our numerical example, we can conclude that our solution holds the potential to improve energy consumption by 24%, bolster job opportunities by 67%, and decrease lost workdays by 18%. Crucially, our sensitivity analyses offered valuable insights into the relationship between sustainability criteria and the optimality of the objective function. It was evident that, except for the number of workers employed, an increase in sustainability criteria directly improved solution optimality, underlining the significance of these criteria in scheduling decisions (Figure 5.12). Based on our results, we can conclude the following managerial insights:

- The adoption of scenario-based robust optimization models, such as the one showcased here, holds great potential for enhancing manufacturing operations. These models are instrumental in addressing uncertainties and variations commonly encountered in real-world production scenarios, ultimately resulting in the generation of more resilient schedules. In particular, the scenario-based robust optimization model we propose empowers production planners to account for both the average and standard deviations in expected makespan, providing a comprehensive view of scheduling reliability.
- To optimize the performance of the ALNS algorithm, it is essential to calibrate parameter adjustments according to the specific characteristics and scale of the production scheduling problem under consideration. To achieve this, production managers and planners can employ calibration techniques, with a particular focus on utilizing the RPD metric. Managers should maintain continuous vigilance over the RPD, aiming to find the optimal balance where solution quality remains satisfactory while computational resources are used efficiently.
- For an in-depth evaluation of the ALNS algorithm's validation, it is crucial to assess the optimality gap as an indicator of solution quality. For smaller-scale problems, we can employ an exact solver, while for larger-scale ones, we can utilize a lower bound based on Lagrangian relaxation or Benders decomposition to estimate the optimal solution. By comparing the solutions produced by ALNS

to this reference, we can estimate the optimality gap. A smaller optimality gap indicates a closer approximation to the optimal solution. Consistently monitoring this metric ensures that the solutions consistently meet the desired quality standards.

- Managers should carefully balance sustainability criteria in production scheduling. While striving to reduce energy consumption and workforce requirements, they should also evaluate the impact of such factors on the makespan to avoid suboptimal schedules.

In conclusion, our research not only advances the field of production scheduling but also provides practical insights for managers seeking to optimize their scheduling processes while considering sustainability and robustness in a dynamic manufacturing landscape.

5.7 Conclusions and future works

In pursuit of addressing the SDPFSP while simultaneously considering economic, environmental, and social criteria, this study undertook a transformative journey. Our primary objective was to devise an operationally robust and efficient solution that could adapt to the disruptions inherent in real-world manufacturing scenarios. To this end, we embarked on a multifaceted approach. We reformulated the SDPFSP into a robust scenario-based optimization model, encompassing the complexities of varying machine operating modes, energy consumption levels, workforce considerations, and the potential for disruptive events such as random job arrivals, machine breakdowns, and uncertain processing times.

At the core of our solution strategy is the development of the ALNS metaheuristic algorithm. This algorithm employed an intricate interplay of four construction and six removal heuristics, augmented by a local search algorithm that incorporated simulated annealing as a decision rule for probabilistic solution acceptance and rejection. The overarching conclusion of our study underscores the effectiveness of the developed ALNS. It consistently demonstrated superior performance in approaching optimal solutions compared to other neighborhood-based metaheuristics, namely SA and VNS. This robustness and efficiency were evident across a spectrum of problem instances, ranging from small to medium to large scales.

In our pursuit of fine-tuning our algorithm, we conducted a sensitivity analysis on the heuristics employed within ALNS. This analysis revealed the efficacy of the R4 removal heuristic, which is based on machine makespan, in combination with C1, a combination of decision rules NR1 and AF1. It became evident that removal heuristics play a pivotal role in the exploration of novel solutions. Furthermore, among the removal heuristics, R4 emerged as the most efficient, while C4, combining decision rules NR2 and AF2, excelled as the best construction heuristic.

Turning our focus towards sustainability criteria, we examined economic, environmental, and social factors, including financial budget, maximum energy consumption by machines, minimum workforce requirements, and the maximum allowable number of lost workdays. Our case study, which revolved around the production of a flange, revealed critical insights. After solving this numerical example, we can conclude that our solution holds the potential to improve energy consumption by 24%, bolster job opportunities by 67%, and decrease lost workdays by 18%. It also became evident that augmenting the budget, increasing the maximum allowable energy consumption, and extending the leeway for lost workdays can lead to enhancements in solution quality. However, it is imperative to note that increasing the minimum number of employed workers hurt solution quality.

In conclusion, this research has delivered a robust and operationally efficient framework for tackling the SDPFSP, marked by the introduction of the custom-designed ALNS, a pioneering solution in the field. However, the journey is far from over. Future research should explore real-time rescheduling strategies and policies to further enhance the SDPFSP solution using our robust optimization approach. Additionally, refining the proposed ALNS through the integration of adaptive memory and tabu list-based approaches for the selection of removal-construction heuristic pairs stands as a promising avenue for further investigation.

CONCLUSIONS AND RECOMMENDATIONS

This thesis encompasses various aspects of sustainable production scheduling in dynamic environments. In Chapter 1, we laid the foundation by outlining the objectives, research questions, and methodology employed throughout the study. Chapter 2 provided an overview of the existing models and studies contributing to uncertain production scheduling, sustainable production scheduling, and the Distributed Permutation Flow Shop Scheduling Problem (DPFSP) while identifying key research gaps and the contributions of this thesis.

To recall our first research objective for the development of a deterministic model for the sustainable DPFSP, Chapter 3 focused on the development of multi-objective optimization model for the sustainable DPFSP that integrated economic, environmental, and social criteria. The results demonstrated the viability of the proposed model, highlighting the advantages of considering different production centers, operating modes, and social factors. Recommendations were made to further investigate uncertainty factors, incorporate risk considerations, and apply the TBL concept to other production scheduling problems. Additionally, the importance of parameter settings and the implementation of the triple bottom line concept in production systems were emphasized.

To recall our second research objective for the development of a smart and sustainable DPFSP using real-time scheduling, Chapter 4 developed an online mixed integer programming model integrating sustainability and uncertainty considerations with real-time events. The study showcased the effectiveness of the proposed model in minimizing makespan while addressing energy consumption, job opportunities, and working days lost. Real-time scheduling strategies and policies were evaluated, with predictive-reactive scheduling and event-driven rescheduling identified as superior approaches. The limitations identified in the study pointed toward future research directions, including robust optimization, multi-objective optimization, and the application of local search metaheuristics to enhance solution quality and efficiency.

To recall our last research objective for the development of a smart and sustainable DPFSP using scenario-based robust optimization, Chapter 5 shifted the focus toward the robust scenario-based optimization of the Sustainable DPFSP (SDPFSP). The developed ALNS algorithm outperformed other metaheuristics in approaching the optimal solution. The study highlighted the effectiveness of the ALNS algorithm and provided insights into the performance of individual construction and removal heuristics. An industrial numerical example validated the applicability of the SDPFSP model, while sensitivity analyses emphasized the impact of sustainability criteria on solution optimality. Recommendations for future research included exploring real-time rescheduling strategies, further enhancing the ALNS algorithm, and considering adaptive memory-based approaches.

This thesis provides a wealth of managerial insights tailored to address the multifaceted challenges encountered by manufacturing managers in today's ever-evolving production landscapes. Central to these insights is the imperative integration of sustainability criteria, spanning economic, environmental, and social aspects, into the realm of production scheduling. It serves as a resounding reminder that efficient scheduling should never come at the expense of workforce well-being or broader societal impact. Manufacturing managers are encouraged to embrace a holistic perspective, ensuring their decisions harmonize with overarching sustainability objectives. Maintaining equilibrium among these sustainability criteria stands out as a pivotal managerial responsibility. While striving to curtail energy consumption and workforce demands, managers should judiciously assess their implications on makespan to safeguard schedule efficiency. A noteworthy stride lies in the inclusion of social factors, such as job opportunities and mitigating lost working days. This addition underscores the significance of striking a balance between economic efficiency and social objectives. It emphasizes that responsible scheduling transcends production optimization, extending to the nurturing of a workforce and minimizing societal disruptions.

Furthermore, the introduction of advanced optimization techniques, exemplified by the multi-objective learning-based SEO algorithm and our ALNS metaheuristic, spotlights the untapped potential of sophisticated tools in decision-making. Manufacturing managers are encouraged to delve into these techniques, recognizing their capacity to elevate production scheduling processes and bring sustainability objectives within reach.

Sensitivity analysis underscores the critical importance of adaptability in scheduling. Managers must remain attuned to the influential role played by key parameters, including budget constraints, environmental considerations, and social weights, in shaping scheduling outcomes. This awareness equips them with the agility needed to fine-tune strategies, ensuring efficiency while steadfastly pursuing sustainability goals.

Redefining task scheduling to accommodate real-time events and uncertainties emerges as a necessity. The choice of scheduling strategy (predictive-reactive or proactive-reactive) and rescheduling policy (continuous or event-driven) holds substantial sway over scheduling efficiency. Hence, managers are urged to meticulously evaluate these choices to align with the unique dynamics of their production environments. Additionally, the thesis underlines the efficiency and robustness of various scheduling methods, encompassing heuristics and reformulations. Manufacturing managers should seriously consider integrating these methods into their repertoire to streamline computational time and unearth solutions that adeptly meet the challenges of scheduling. Finally, the spotlight falls on robust optimization models, particularly scenario-based ones, as invaluable tools. These models empower managers to navigate the intricate web of uncertainties and variations inherent in scheduling, ultimately yielding schedules resilient enough to withstand real-world disruptions.

Based on above findings and recommendations, we can address our research questions as follows:

- How can the principles of sustainability be integrated into the optimization models and methodologies for the DPFSP?

As discussed in Chapter 3, a multi-objective optimization model can be developed. This model should consider economic, environmental, and social criteria simultaneously. Sustainability objectives, such as minimizing energy consumption, reducing environmental impact, and maximizing job opportunities, should be formulated as objectives or constraints within the optimization model. In addition, as discussed in Chapter 4, we should have a careful attention to parameter settings within the optimization model. Parameters related to sustainability criteria (e.g., the budget of the company, the bounds of the energy consumption and social factors) should be adjusted to reflect the organization's sustainability goals and priorities. Finally, as highlighted in the thesis, sensitivity analysis is crucial for integrating sustainability into the DPFSP. Managers should conduct sensitivity analyses to understand how changes in parameters, constraints, or objectives impact the sustainability of schedules. This helps in making informed decisions that balance efficiency and sustainability.

- How can real-time scheduling strategies and policies be effectively employed in the DPFSP to handle uncertainties and disruptions?

As Chapter 4 suggests, real-time scheduling involves the application of two primary strategies: predictive-reactive and proactive-reactive scheduling. To enable predictive-reactive scheduling, it is essential for managers to allocate resources to the development and deployment of predictive analytics tools and algorithms. These tools can provide early warnings and insights into potential disruptions in the production process. By leveraging predictive capabilities, managers can make informed decisions to mitigate disruptions before they occur.

Real-time scheduling necessitates continuous monitoring of production processes. Managers should proactively invest in technologies such as IoT devices and sensor systems. These technologies enable the real-time collection of data pertaining to machine status, inventory levels, and other relevant production factors. This data serves as the foundation for making timely and data-driven scheduling decisions.

In conjunction with continuous monitoring, managers should have well-defined rescheduling policies in place. These policies encompass both continuous and event-driven rescheduling approaches. Continuous rescheduling involves making periodic adjustments to the production schedule based on real-time data, while event-driven rescheduling entails immediate responses to unexpected events or disruptions.

- How can scenario-based robust optimization techniques be employed to improve the robustness and resilience of the DPFSP schedules?

To effectively address uncertainties, the utilization of robust optimization techniques is advisable, as expounded upon in Chapter 5. These techniques are instrumental in crafting schedules that exhibit resilience in the face of diverse disruptions, achieved by factoring in multiple scenarios. This entails the exploration of various scenarios that encapsulate uncertainties such as job processing times and machine breakdowns, followed by the optimization of schedules to bolster their robustness within this spectrum of scenarios.

Furthermore, validating the robust optimization model with real-world industrial data is of paramount importance. Manufacturing managers should actively engage in collaboration with researchers and industry practitioners to ensure that the scenarios incorporated into the model faithfully mirror the challenges encountered in actual production settings. This validation process enhances the practical applicability and reliability of the optimization approach.

- How can advanced metaheuristics be leveraged to optimize the DPFSP schedules?

As emphasized in the thesis, there is a valuable opportunity to harness advanced metaheuristics, such as the SEO algorithm, to enhance scheduling within the DPFSP. Manufacturing managers are encouraged to explore the application of these algorithms to optimize DPFSP schedules. These metaheuristics excel at efficiently navigating complex scheduling spaces, enabling the discovery of high-quality solutions. Additionally, as highlighted in Chapter 5, the ALNS algorithm stands out as a formidable tool. This algorithm offers the capability to optimize schedules by systematically considering a variety of removal and construction heuristics, thus facilitating the exploration of diverse manufacturing solutions.

Moreover, the thesis proposes the integration of local search metaheuristics to augment solution quality and operational efficiency. Managers should contemplate the incorporation of these local search techniques within their optimization algorithms, allowing for the fine-tuning of schedules to better align with specific production requirements. In the broader context of employing metaheuristics, it is crucial to consider computational efficiency. Managers must rigorously assess the computational resources demanded by these algorithms to ensure that they remain within acceptable limits for practical implementation. Balancing computational efficiency with solution quality is pivotal in realizing the benefits of metaheuristic approaches within the DPFSP and similar scheduling challenges.

In conclusion, this Ph.D. project significantly contributes to the field of sustainable production scheduling by addressing the complex challenges of integrating economic, environmental, and social criteria in dynamic production environments. The findings demonstrated the effectiveness of the proposed models and algorithms in optimizing various objectives while considering real-time events, uncertainty, and sustainability criteria. However, several avenues for future research remain, including the exploration of robust and stochastic optimization concepts, the incorporation of risk factors, the application of multi-objective optimization techniques, and the utilization of local search metaheuristics. Furthermore, implementing real-time rescheduling strategies and refining the proposed algorithms can enhance solution quality and efficiency. Pursuing these research directions will enable improvements in the sustainability and operational performance of production systems across diverse industries.

APPENDIX I

ACADEMIC ACCOMPLISHMENTS

Journal Articles

1- **Submitted**

Fathollahi-Fard, A. M., Woodward, L., & Akhrif, O. A (2023). Distributed permutation flow shop considering sustainability criteria and real-time scheduling, Journal of Industrial Information Integration, Under review (R1)

Fathollahi-Fard, A. M., Woodward, L., & Akhrif, O. A (2023). A robust optimization model for the sustainable distributed permutation flow-shop scheduling problem, Annals of Operations Research, Under review (R1)

Fathollahi-Fard, A. M., Woodward, L., & Akhrif, O. (2023). An optimization model for smart and sustainable distributed permutation flow-shop scheduling. International Conference on Industry 4.0 and Smart Manufacturing, Accepted

2- **Published**

Fathollahi-Fard, A. M., Woodward, L., & Akhrif, O. (2021). Sustainable distributed permutation flow-shop scheduling model based on a triple bottom line concept. Journal of Industrial Information Integration, 24, 100233.

BIBLIOGRAPHY

Abreu, L. R., Cunha, J. O., Prata, B. A., & Framinan, J. M. (2020). A genetic algorithm for scheduling open shops with sequence-dependent setup times. *Computers & Operations Research*, 113, 104793.

Al-Behadili, M., Ouelhadj, D., & Jones, D. (2020). Multi-objective biased randomised iterated greedy for robust permutation flow shop scheduling problem under disturbances. *Journal of the Operational Research Society*, *71*(11), 1847-1859.

Alphonsus, E. R., & Abdullah, M. O. (2016). A review on the applications of programmable logic controllers (PLCs). *Renewable and Sustainable Energy Reviews*, *60*, 1185-1205.

Anghinolfi, D., Paolucci, M., & Ronco, R. (2020). A bi-objective heuristic approach for green identical parallel machine scheduling. *European Journal of Operational Research*, *289*(2), 416-434.

Bargaoui, H., Driss, O. B., & Ghédira, K. (2017). A novel chemical reaction optimization for the distributed permutation flowshop scheduling problem with makespan criterion. *Computers & Industrial Engineering*, *111*, 239-250.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, *4*(1), 238-252.

Bellio, R., Ceschia, S., Di Gaspero, L., & Schaerf, A. (2021). Two-stage multi-neighborhood simulated annealing for uncapacitated examination timetabling. *Computers & Operations Research*, *132*, 105300.

Benoît, C., Norris, G. A., Valdivia, S., Ciroth, A., Moberg, A., Bos, U., & Beck, T. (2010). The guidelines for social life cycle assessment of products: just in time! *The International Journal of Life Cycle Assessment*, *15*(2), 156-163.

Błażewicz, J., Domschke, W., & Pesch, E. (1996). The job shop scheduling problem: Conventional and new solution techniques. *European Journal of Operational Research*, *93*(1), 1-33.

Che, A., Wu, X., Peng, J., & Yan, P. (2017). Energy-efficient bi-objective single-machine scheduling with power-down mechanism. *Computers & Operations Research*, *85*, 172-183.

Cheraghalipour, A., Paydar, M. M., & Hajiaghaei-Keshteli, M. (2018). A bi-objective optimization for citrus closed-loop supply chain using Pareto-based algorithms. *Applied Soft Computing*, *69*, 33-59.

Corbett, C. J., & Kirsch, D. A. (2001). International diffusion of ISO 14000 certification. *Production and Operations Management*, *10*(3), 327-342.

Conti, J., Holtberg, P., Diefenderfer, J., LaRose, A., Turnure, J. T., & Westfall, L. (2016). *International energy outlook 2016 with projections to 2040* (No. DOE/EIA-0484 (2016)). USDOE Energy Information Administration (EIA), Washington, DC (United States). Office of Energy Analysis.

Dalenogare, L. S., Benitez, G. B., Ayala, N. F., & Frank, A. G. (2018). The expected contribution of Industry 4.0 technologies for industrial performance. *International Journal of Production Economics*, *204*, 383-394.

Dai, M., Tang, D., Giret, A., & Salido, M. A. (2019). Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints. *Robotics and Computer-Integrated Manufacturing*, *59*, 143-157.

Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *International conference on parallel problem solving from nature* (pp. 849-858). Springer, Berlin, Heidelberg.

Dolgui, A., Ivanov, D., Sethi, S. P., & Sokolov, B. (2019). Scheduling in production, supply chain and Industry 4.0 systems by optimal control: fundamentals, state-of-the-art and applications. *International Journal of Production Research*, *57*(2), 411-432.

Duarte, J. L. R., Fan, N., & Jin, T. (2020). Multi-process production scheduling with variable renewable integration and demand response. *European Journal of Operational Research*, 281(1), 186-200.

Fard, A. M. F., & Hajaghaei-Keshteli, M. (2018). A tri-level location-allocation model for forward/reverse supply chain. *Applied Soft Computing*, *62*, 328-346.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., & Tavakkoli-Moghaddam, R. (2018). The social engineering optimizer (SEO). *Engineering Applications of Artificial Intelligence*, *72*, 267-293.

Fathollahi-Fard, A. M., Ranjbar-Bourani, M., Cheikhrouhou, N., & Hajiaghaei-Keshteli, M. (2019). Novel modifications of social engineering optimizer to solve a truck scheduling problem in a cross-docking system. *Computers & Industrial Engineering*, *137*, 106103.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., Tian, G., & Li, Z. (2020). An adaptive Lagrangian relaxation-based algorithm for a coordinated water supply and wastewater collection network design problem. *Information Sciences*, *512*, 1335-1359.

Fathollahi-Fard, A. M., Ahmadi, A., & Mirzapour Al-e-Hashem, S. M. J., (2020). Sustainable Closed-loop Supply Chain Network for an Integrated Water Supply and Wastewater Collection System under Uncertainty, *Journal of Environmental Management*, 275, 111277.

Fathollahi-Fard, A. M., Woodward, L., & Akhrif, O. (2021). Sustainable distributed permutation flow-shop scheduling model based on a triple bottom line concept. *Journal of Industrial Information Integration*, 100233.

Fathollahi-Fard, A. M., Ahmadi, A., & Karimi, B. (2022). Sustainable and robust home healthcare logistics: A response to the covid-19 pandemic. *Symmetry*, *14*(2), 193.

Fisher, M. L. (1981). The Lagrangian relaxation method for solving integer programming problems. *Management Science*, *27*(1), 1-18.

Fu, Y., Ding, J., Wang, H., & Wang, J. (2018). Two-objective stochastic flow-shop scheduling with deteriorating and learning effect in Industry 4.0-based manufacturing system. *Applied Soft Computing*, *68*, 847-855.

Fu, Y., Tian, G., Fathollahi-Fard, A. M., Ahmadi, A., & Zhang, C. (2019). Stochastic multi-objective modelling and optimization of an energy-conscious distributed permutation flow shop scheduling problem with the total tardiness constraint. *Journal of Cleaner Production*, *226*, 515-525.

Fernandez-Viagas, V., Perez-Gonzalez, P., & Framinan, J. M. (2018). The distributed permutation flow shop to minimise the total flowtime. *Computers & Industrial Engineering*, *118*, 464-477.

Frank, A. G., Dalenogare, L. S., & Ayala, N. F. (2019). Industry 4.0 technologies: Implementation patterns in manufacturing companies. *International Journal of Production Economics*, *210*, 15-26.

Framinan, J. M., Fernandez-Viagas, V., & Perez-Gonzalez, P. (2019). Using real-time information to reschedule jobs in a flowshop with variable processing times. *Computers & Industrial Engineering*, *129*, 113-125.

Gahm, C., Denz, F., Dirr, M., & Tuma, A. (2016). Energy-efficient scheduling in manufacturing companies: A review and research framework. *European Journal of Operational Research*, *248*(3), 744-757.

Gao, J., & Chen, R. (2011). A hybrid genetic algorithm for the distributed permutation flowshop scheduling problem. *International Journal of Computational Intelligence Systems*, *4*(4), 497-508.

Gao, K. Z., Suganthan, P. N., Chua, T. J., Chong, C. S., Cai, T. X., & Pan, Q. K. (2015). A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion. *Expert systems with applications*, *42*(21), 7652-7663.

Ghaleb, M., Zolfagharinia, H., & Taghipour, S. (2020). Real-time production scheduling in the Industry-4.0 context: Addressing uncertainties in job arrivals and machines breakdowns. *Computers & Operations Research*, 105031.

Gholizadeh, H., Fazlollahtabar, H., Fathollahi-Fard, A. M., & Dulebenets, M. A. (2021). Preventive maintenance for the flexible flowshop scheduling under uncertainty: a waste-to-energy system. *Environmental Science and Pollution Research*, 1-20.

Graves, S. C. (1981). A review of production scheduling. *Operations research*, *29*(4), 646-675.

Gmys, J., Mezmaz, M., Melab, N., & Tuyttens, D. (2020). A computationally efficient branch-and-bound algorithm for the permutation flow-shop scheduling problem. *European Journal of Operational Research*, *284*(3), 814-833.

Gonçalves, J. F., de Magalhães Mendes, J. J., & Resende, M. G. (2005). A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operational Research*, *167*(1), 77-95.

Gonçalves, J. F., & Resende, M. G. (2011). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, *17*(5), 487-525.

Han, Y., Li, J. Q., Gong, D., & Sang, H. (2018). Multi-objective migrating birds optimization algorithm for stochastic lot-streaming flow shop scheduling with blocking. *IEEE Access*, *7*, 5946-5962.

Han, Y., Li, J., Sang, H., Liu, Y., Gao, K., & Pan, Q. (2020). Discrete evolutionary multi-objective optimization for energy-efficient blocking flow shop scheduling with setup time. *Applied Soft Computing*, 106343.

Harmonosky, C. M., & Robohn, S. F. (1991). Real-time scheduling in computer integrated manufacturing: a review of recent research. *International Journal of Computer Integrated Manufacturing*, *4*(6), 331-340.

Hamzadayı, A. (2020). An effective benders decomposition algorithm for solving the distributed permutation flowshop scheduling problem. *Computers & Operations Research*, *123*, 105006.

Haimes, Y.Y., Ladson, L.S., & Wismer, D.A., (1971). Bicriterion formulation of problems of integrated system identification and system optimization. *IEEE Transaction on Systems, Man and Cybernetic*. 296 (3), 34-41.

He, W., & Sun, D. H. (2013). Scheduling flexible job shop problem subject to machine breakdown with route changing and right-shift strategies. *The International Journal of Advanced Manufacturing Technology*, *66*(1-4), 501-514.

Hong, Z., Zeng, Z., & Gao, L. (2021). Energy-efficiency scheduling of multi-cell manufacturing system considering total handling distance and eligibility constraints. *Computers & Industrial Engineering*, *151*, 106998.

Huang, J., & Gu, X. (2021). Distributed assembly permutation flow-shop scheduling problem with sequence-dependent set-up times using a novel biogeography-based optimization algorithm. *Engineering Optimization*, 145, 1-21.

Huang, Y. Y., Pan, Q. K., Huang, J. P., Suganthan, P. N., & Gao, L. (2021). An improved iterated greedy algorithm for the distributed assembly permutation flowshop scheduling problem. *Computers & Industrial Engineering*, *152*, 107021.

Karna, S. K., & Sahai, R. (2012). An overview on Taguchi method. *International Journal of Engineering and Mathematical Sciences*, *1*(1), 1-7.

Karmaker, C. L., Ahmed, T., Ahmed, S., Ali, S. M., Moktadir, M. A., & Kabir, G. (2020). Improving supply chain sustainability in the context of COVID-19 pandemic in an emerging economy: Exploring drivers using an integrated model. *Sustainable Production and Consumption*, *26*, 411-427.

Jiang, X., Tian, Z., Liu, W., Suo, Y., Chen, K., Xu, X., & Li, Z. (2022). Energy-efficient scheduling of flexible job shops with complex processes: A case study for the aerospace industry complex components in China. *Journal of Industrial Information Integration*, *27*, 100293.

Jing, X. L., Pan, Q. K., & Gao, L. (2021). Local search-based metaheuristics for the robust distributed permutation flowshop problem. *Applied Soft Computing*, 107247.

Jiménez, M., Arenas, M., Bilbao, A., & Rodrı, M. V. (2007). Linear programming with fuzzy parameters: an interactive method resolution. *European journal of operational research*, *177*(3), 1599-1609.

Llach, J., Marimon, F., & del Mar Alonso-Almeida, M. (2015). Social Accountability 8000 standard certification: analysis of worldwide diffusion. *Journal of Cleaner Production*, *93*, 288-298.

Leung, S. C., Tsang, S. O., Ng, W. L., & Wu, Y. (2007). A robust optimization model for multi-site production planning problem in an uncertain environment. *European Journal of Operational Research*, *181*(1), 224-238.

Li, C., Li, L., Tang, Y., Zhu, Y., & Li, L. (2019). A comprehensive approach to parameters optimization of energy-aware CNC milling. *Journal of Intelligent Manufacturing*, *30*(1), 123-138.

Li, J. Q., Pan, Q. K., & Mao, K. (2015). A hybrid fruit fly optimization algorithm for the realistic hybrid flowshop rescheduling problem in steelmaking systems. *IEEE Transactions on Automation Science and Engineering*, *13*(2), 932-949.

Li, D. C., & Hsu, P. H. (2012). Solving a two-agent single-machine scheduling problem considering learning effect. *Computers & Operations Research*, *39*(7), 1644-1651.

Lin, S. W., Ying, K. C., & Huang, C. Y. (2013). Minimising makespan in distributed permutation flowshops using a modified iterated greedy algorithm. *International Journal of Production Research*, *51*(16), 5029-5038.

Liu, F., Wang, S., Hong, Y., & Yue, X. (2017a). On the robust and stable flowshop scheduling under stochastic and dynamic disruptions. *IEEE Transactions on Engineering Management*, *64*(4), 539-553.

Liu, X., Laporte, G., Chen, Y., & He, R. (2017b). An adaptive large neighborhood search metaheuristic for agile satellite scheduling with time-dependent transition time. *Computers & Operations Research*, *86*, 41-53.

Lu, C., Gao, L., Gong, W., Hu, C., Yan, X., & Li, X. (2020). Sustainable Scheduling of Distributed Permutation Flow-shop with Non-identical Factory Using A Knowledge-based Multi-objective Memetic Optimization Algorithm. *Swarm and Evolutionary Computation*, 100803.

Lu, C., Gao, L., Li, X., & Chen, P. (2016). Energy-efficient multi-pass turning operation using multi-objective backtracking search algorithm. *Journal of Cleaner Production*, 137, 1516-1531.

Manríquez, F., González, H., & Morales, N. (2022). Short-term open-pit production scheduling optimizing multiple objectives accounting for shovel allocation in stockpiles. *Optimization and Engineering*, 1-27.

Manne, A. S. (1960). On the job-shop scheduling problem. *Operations Research*, *8*(2), 219-223.

Mansouri, S. A., Aktas, E., & Besikci, U. (2016). Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption. *European Journal of Operational Research*, *248*(3), 772-788.

Marichelvam, M. K., & Geetha, M. (2021). A memetic algorithm to solve uncertain energy-efficient flow shop scheduling problems. *The International Journal of Advanced Manufacturing Technology*, *115*(1-2), 515-530.

Mara, S. T. W., Norcahyo, R., Jodiawan, P., Lusiantoro, L., & Rifai, A. P. (2022). A survey of adaptive large neighborhood search algorithms and applications. *Computers & Operations Research*, 105903.

Marchi, B., & Zanoni, S. (2017). Supply chain management for improved energy efficiency: Review and opportunities. *Energies*, *10*(10), 1618.

Marimuthu, R., Sankaranarayanan, B., Ali, S. M., de Sousa Jabbour, A. B. L., & Karuppiah, K. (2021). Assessment of key socio-economic and environmental challenges in the mining industry: Implications for resource policies in emerging economies. *Sustainable Production and Consumption*. 27, 814-830.

Mehta, S. V., & Uzsoy, R. M. (1998). Predictable scheduling of a job shop subject to breakdowns. *IEEE Transactions on Robotics and Automation*, *14*(3), 365-378.

Meng, T., Pan, Q. K., & Wang, L. (2019). A distributed permutation flowshop scheduling problem with the customer order constraint. *Knowledge-Based Systems*, *184*, 104894.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & operations research*, *24*(11), 1097-1100.

Mokhtari, H., & Hasani, A. (2017). An energy-efficient multi-objective optimization for flexible job-shop scheduling problem. *Computers & Chemical Engineering*, *104*, 339-352.

Mojtahedi, M., Fathollahi-Fard, A. M., Tavakkoli-Moghaddam, R., & Newton, S. (2021). Sustainable vehicle routing problem for coordinated solid waste management. *Journal of Industrial Information Integration*, *23*, 100220.

Mulvey, J. M., Vanderbei, R. J., & Zenios, S. A. (1995). Robust optimization of large-scale systems. *Operations research*, *43*(2), 264-281.

Naderi, B., & Ruiz, R. (2010). The distributed permutation flowshop scheduling problem. *Computers & Operations Research*, *37*(4), 754-768.

Naderi, B., & Ruiz, R. (2014). A scatter search algorithm for the distributed permutation flowshop scheduling problem. *European Journal of Operational Research*, *239*(2), 323-334.

Pan, Q. K., Gao, L., Wang, L., Liang, J., & Li, X. Y. (2019). Effective heuristics and metaheuristics to minimize total flowtime for the distributed permutation flowshop problem. *Expert Systems with Applications*, *124*, 309-324.

Pasha, J., Nwodu, A. L., Fathollahi-Fard, A. M., Tian, G., Li, Z., Wang, H., & Dulebenets, M. A. (2022). Exact and metaheuristic algorithms for the vehicle routing problem with a factory-in-a-box in multi-objective settings. *Advanced Engineering Informatics*, *52*, 101623.

Parente, M., Figueira, G., Amorim, P., & Marques, A. (2020). Production scheduling in the context of Industry 4.0: Review and trends. *International Journal of Production Research*, *58*(17), 5401-5431.

Rahmani, D., & Ramezanian, R. (2016). A stable reactive approach in dynamic flexible flow shop scheduling with unexpected disruptions: A case study. *Computers & Industrial Engineering*, *98*, 360-372.

Rahmaniani, R., Crainic, T. G., Gendreau, M., & Rei, W. (2017). The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, *259*(3), 801-817.

Rossit, D. A., Tohmé, F., & Frutos, M. (2019a). Industry 4.0: smart scheduling. *International Journal of Production Research*, *57*(12), 3802-3813.

Rossit, D. A., Tohme, F., & Frutos, M. (2019b). Production planning and scheduling in Cyber-Physical Production Systems: a review. *International Journal of Computer Integrated Manufacturing*, *32*(4-5), 385-395.

Rossit, D., & Tohmé, F. (2018). Scheduling research contributions to Smart manufacturing. *Manufacturing Letters*, *15*, 111-114.

Ross, M. S. (2019). Introduction to Probability Models. Elsevier books.

Rothe, I., Susse, H., & Voss, K. (1996). The method of normalization to determine invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(4), 366-376.

Roy, R. K. (2010). *A primer on the Taguchi method*. Society of Manufacturing Engineers.

Ruiz, R., Pan, Q. K., & Naderi, B. (2019). Iterated Greedy methods for the distributed permutation flowshop scheduling problem. *Omega*, *83*, 213-222.

Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research*, *177*(3), 2033-2049.

Sabuj, S. U., Ali, S. M., Hasan, K. W., & Paul, S. K. (2021). Contextual relationships among key factors related to environmental sustainability: Evidence from an emerging economy. *Sustainable Production and Consumption*, *27*, 86-99.

Shahrabi, J., Adibi, M. A., & Mahootchi, M. (2017). A reinforcement learning approach to parameter estimation in dynamic job shop scheduling. *Computers & Industrial Engineering*, *110*, 75-82.

Shao, W., Shao, Z., & Pi, D. (2020). Modeling and multi-neighborhood iterated greedy algorithm for distributed hybrid flow shop scheduling problem. *Knowledge-Based Systems*, 105527.

Shen, X. N., & Yao, X. (2015). Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems. *Information Sciences*, *298*, 198-224.

Shilyaev, P. V., Andryushin, I. Y., Golovin, V. V., Radionov, A. A., Karandaev, A. S., & Khramshin, V. R. (2013). Algorithms of a digital automatic system for tension and loop control in a wide-strip hot-rolling mill. *Russian Electrical Engineering*, *84*(10), 533-541.

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., & Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, *159*(2), 139-171.

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *International conference on principles and practice of constraint programming* (pp. 417-431). Springer, Berlin, Heidelberg.

Shukla, A. K., Nath, R., Muhuri, P. K., & Lohani, Q. D. (2020). Energy efficient multi-objective scheduling of tasks with interval type-2 fuzzy timing constraints in an Industry 4.0 ecosystem. *Engineering Applications of Artificial Intelligence*, *87*, 103257.

Sin, I. H., & Do Chung, B. (2020). Bi-objective optimization approach for energy aware scheduling considering electricity cost and preventive maintenance using genetic algorithm. *Journal of Cleaner Production*, *244*, 118869.

Soleimani, H., Chhetri, P., Fathollahi-Fard, A. M., Mirzapour Al-e-Hashem, S. M. J., & Shahparvari, S. (2022). Sustainable closed-loop supply chain with energy efficiency: Lagrangian relaxation, reformulations and heuristics. *Annals of Operations Research*, *318*(1), 531-556.

Tang, L., Liu, J., Rong, A., & Yang, Z. (2001). A review of planning and scheduling systems and methods for integrated steel production. *European Journal of Operational Research*, *133*(1), 1-20.

Tautenhain, C. P., Barbosa-Povoa, A. P., Mota, B., & Nascimento, M. C. (2021). An efficient Lagrangian-based heuristic to solve a multi-objective sustainable supply chain problem. *European Journal of Operational Research*, *294*(1), 70-90.

Tordecilla, R. D., Montoya-Torres, J. R., Quintero-Araujo, C. L., Panadero, J., & Juan, A. A. (2023). The location routing problem with facility sizing decisions. *International Transactions in Operational Research*, *30*(2), 915-945.

Tirkolaee, E. B., Goli, A., & Weber, G. W. (2020). Fuzzy mathematical programming and self-adaptive artificial fish swarm algorithm for just-in-time energy-aware flow shop scheduling problem with outsourcing option. *IEEE Transactions on Fuzzy Systems*, *28*(11), 2772-2783.

Ullah, I., Zhang, S., Zhang, Q., & Wang, R. (2021). Numerical investigation on serrated chip formation during high-speed milling of Ti-6Al-4V alloy. *Journal of Manufacturing Processes*, *71*, 589-603.

Varelmann, T., Erwes, N., Schäfer, P., & Mitsos, A. (2022). Simultaneously optimizing bidding strategy in pay-as-bid-markets and production scheduling. *Computers & Chemical Engineering*, *157*, 107610.

Van Laarhoven, P. J., & Aarts, E. H. (1987). Simulated annealing. In *Simulated annealing: Theory and applications* (pp. 7-15). Springer, Dordrecht.

Wang, J. J., & Wang, L. (2018). A knowledge-based cooperative algorithm for energy-efficient scheduling of distributed flow-shop. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, (99), 1-15.

Wang, G., Gao, L., Li, X., Li, P., & Tasgetiren, M. F. (2020). Energy-efficient distributed permutation flow shop scheduling problem using a multi-objective whale swarm algorithm. *Swarm and Evolutionary Computation*, 100716.

Wang, S., Wang, X., Yu, J., Ma, S., & Liu, M. (2018). Bi-objective identical parallel machine scheduling to minimize total energy consumption and makespan. *Journal of Cleaner Production*, *193*, 424-440.

Wilson, J. M. (1989). Alternative formulations of a flow-shop scheduling problem. *Journal of the Operational Research Society*, *40*(4), 395-399.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67-82.

Wu, X., & Che, A. (2019). A memetic differential evolution algorithm for energy-efficient parallel machine scheduling. *Omega*, *82*, 155-165.

Wu, X., & Sun, Y. (2018). A green scheduling algorithm for flexible job shop with energy-saving measures. *Journal of Cleaner Production*, *172*, 3249-3264.

Xiong, H., Shi, S., Ren, D., & Hu, J. (2022). A survey of job shop scheduling problem: The types and models. *Computers & Operations Research*, 105731.

Xu, Y., Yan, C., Liu, H., Wang, J., Yang, Z., & Jiang, Y. (2020). Smart Energy Systems: A Critical Review on Design and Operation Optimization. *Sustainable Cities and Society*, 102369.

Xu, Y., Wang, L., Wang, S., & Liu, M. (2014). An effective hybrid immune algorithm for solving the distributed permutation flow-shop scheduling problem. *Engineering Optimization*, *46*(9), 1269-1283.

Ye, H., Wang, X., & Liu, K. (2020). Adaptive preventive maintenance for flow shop scheduling with resumable processing. *IEEE Transactions on Automation Science and Engineering*, *18*(1), 106-113.

Yoon, H. S., Kim, E. S., Kim, M. S., Lee, J. Y., Lee, G. B., & Ahn, S. H. (2015). Towards greener machine tools–A review on energy saving strategies and technologies. *Renewable and Sustainable Energy Reviews*, *48*, 870-891.

Yue, L., Wang, H., Mumtaz, J., Rauf, M., & Li, Z. (2023). Energy-efficient scheduling of a two-stage flexible printed circuit board flow shop using a hybrid Pareto spider monkey optimisation algorithm. *Journal of Industrial Information Integration*, *31*, 100412.

Zhang, J., Ding, G., Zou, Y., Qin, S., & Fu, J. (2019). Review of job shop scheduling research and its new perspectives under Industry 4.0. *Journal of Intelligent Manufacturing*, *30*(4), 1809-1830.

Zhang, B., Pan, Q. K., Gao, L., Meng, L. L., Li, X. Y., & Peng, K. K. (2019). A three-stage multiobjective approach based on decomposition for an energy-efficient hybrid flow shop scheduling problem. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *50*(12), 4984-4999.

Zhang, Qingfu et Li, Hui. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 2007, 11(6), 712-731

Zhang, B., Pan, Q. K., Gao, L., Li, X. Y., Meng, L. L., & Peng, K. K. (2019). A multiobjective evolutionary algorithm based on decomposition for hybrid flowshop green scheduling problem. *Computers & Industrial Engineering*, *136*, 325-344.

Zhu, G. Y., Ding, C., & Zhang, W. B. (2020). Optimal foraging algorithm that incorporates fuzzy relative entropy for solving many-objective permutation flow shop scheduling problems. *IEEE Transactions on Fuzzy Systems*, *28*(11), 2738-2746.

Zhuang, Z., Li, Y., Sun, Y., Qin, W., & Sun, Z. H. (2022). Network-based dynamic dispatching rule generation mechanism for real-time production scheduling problems with dynamic job arrivals. *Robotics and Computer-Integrated Manufacturing*, *73*, 102261.

Zitzler, Eckart, LAUMANNS, Marco, & Thiele, Lothar (2001). SPEA2: Improving the strength Pareto evolutionary algorithm. *TIK-report*, 103, 34-76.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, *8*(2), 173-195.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Da Fonseca, V. G. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, *7*(2), 117-132.

Zitzler, E. (1999). *Evolutionary algorithms for multiobjective optimization: Methods and applications* (Vol. 63). Ithaca: Shaker.

Zitzler, E., Brockhoff, D., & Thiele, L. (2007). The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration. In *International Conference on Evolutionary Multi-Criterion Optimization* (pp. 862-876). Springer, Berlin, Heidelberg.