Production planning and scheduling problems under energy aspects

Mohammadmohsen Aghelinejad

Thèse
de doctorat
de l'UTT

Mohammadmohsen AGHELINEJAD

Production Planning and Scheduling Problems under Energy Aspects

Champ disciplinaire :
Sciences pour l'Ingénieur

2018TROY0043 **Année 2018**

utt
UNIVERSITÉ DE TECHNOLOGIE
TROYES

THESE

pour l'obtention du grade de

DOCTEUR
de l'UNIVERSITE DE TECHNOLOGIE DE TROYES

EN SCIENCES POUR L'INGENIEUR

Spécialité : OPTIMISATION ET SURETE DES SYSTEMES

présentée et soutenue par

Mohammadmohsen AGHELINEJAD

le 29 novembre 2018

Production Planning and Scheduling Problems under Energy Aspects

JURY

M. A. SIADAT	PROFESSEUR DES UNIVERSITES	Président
M. H. ALLAOUI	PROFESSEUR DES UNIVERSITES	Rapporteur
Mme N. SAUER	PROFESSEURE DES UNIVERSITES	Rapporteure
M. F. FRUGGIERO	ASSISTANT PROFESSOR	Examinateur
M. R. TAVAKKOLI-MOGHADDAM	PROFESSOR	Examinateur
M. F. YALAOUI	PROFESSEUR DES UNIVERSITES	Examinateur
M. Y. OUAZENE	MAITRE DE CONFERENCES	Directeur de thèse
Mme A. YALAOUI	MAITRE DE CONFERENCES - HDR	Directrice de thèse

This thesis has been prepared at

Institut Charles Delaunay (ICD)- Laboratoire d'Optimisation des Systèmes Industriels(LOSI)

☎ +33 (0)3 25 71 76 00
🖷 +33 (0)3 25 71 76 75
✉ secretariat-losi@utt.fr
Web Site http://losi.utt.fr

This PhD research Thesis is dedicated to my dear wife, Elnaz, who has been a constant source of support and encouragement during the challenges of graduate school and life. I am truly thankful for having you in my life. This work is also dedicated to all members of my family, especially to my parents, Abbas and Mehri, who have always loved me unconditionally and whose good examples have taught me to work hard for the things that I aspire to achieve.

Production planning and scheduling problems under energy aspects

Abstract

In the last few years, economic and societal developments have led to a rapid increase in energy consumption. Moreover, in almost every industrial nation, electricity prices, one of the main energy sources used in the manufacturing factories, have been rising continuously. Therefore, improving electric power efficiency and saving electricity plays a very important role in modern industries. Recently, several studies are devoted to integrate energy efficiency into the scheduling problems. However, only few of them considered a multi-states machine-scheduling problem under time varied energy prices. Therefore, this thesis deals with the case of a single machine scheduling problem with different operating states characterized by different speeds and energy consumptions. It aims to provide a complete study to investigate the complexity and the optimization methods of different variant of this problem. For this purpose, a dynamic programming approach based on a finite graph is used to model several problems and establish their complexities. We also address different mathematical formulations and optimization methods to solve the NP-hard variants of this problem. A new heuristic algorithm and a genetic algorithm are proposed for the single machine scheduling problem with multi-states and the Time-Of-Use electricity (TOU) costs. A heuristic algorithm, a genetic algorithm as well as a memetic algorithm are proposed for the single machine scheduling problem with multi-states, multi-speeds, TOU costs and different energy consumptions for the jobs.

Keywords: mathematical optimization, production scheduling, energy consumption, metaheuristics

Résumé

Ces dernières années, les évolutions économiques et sociétales ont entraîné une augmentation rapide de la consommation d'énergie. De plus, dans presque tous les pays industriels, le prix de l'électricité a augmenté continuellement. L'amélioration de l'efficacité énergétique et la réduction de la consommation d'électricité jouent donc un rôle très important dans les industries modernes. Récemment, plusieurs études ont été consacrées à l'intégration de l'efficacité énergétique dans les problèmes d'ordonnancement. Cependant, seuls certains travaux précédents considéraient un problème d'une seule machine multi-états avec différents prix d'énergie. Cette thèse aborde le cas d'un problème d'ordonnancement d'une seule machine avec différents états de fonctionnement caractérisés par différentes vitesses et consommations d'énergie. Cette thèse vise à fournir une étude complète pour étudier la complexité et les méthodes d'optimisation des différentes variantes de ce problème. Dans ce but, une nouvelle approche de programmation dynamique basée sur un graphe fini est utilisée pour modéliser plusieurs problèmes et établir leur complexité. Nous abordons également différentes formulations mathématiques et méthodes d'optimisation pour résoudre les variantes NP-difficile de ce problème. Des méthodes approchées sont proposées pour résoudre le problème d'ordonnancement d'une seule machine avec multi-états et les coûts d'électricité de temps d'utilisation dans le cas avec vitesse unique ainsi que dans le cas avec vitesse multiple et différentes consommations d'énergie pour les tâches.

Mots clés : optimisation mathématique, ordonnancement (gestion), consommation d'énergie, métaheuristiques

Institut Charles Delaunay (ICD)- Laboratoire d'Optimisation des Systèmes Industriels(LOSI)

Acknowledgment

First of all, I would like to thank the members of the jury who gave me the great honour to agree to evaluate this thesis: Mrs. Nathalie SAUER professor at Lorraine University, Mr. Hamid ALLAOUI professor at Artois University, Mr. Farouk YALAOUI professor at University of Technology of Troyes, Mr. Ali SIADAT professor at Ecole Nationale d'Arts et Métier Paris-tech and Mr. Fabio Fruggiero assistant professor at university of degli Studi della Basilicata. I thank them very much for the time they have devoted to this, despite all the responsibilities they have.

My sincere thanks are also dedicated to my supervisors, Dr. Alice YALAOUI and Dr. Yassine OUAZENE. I thank them for the confidence that they have given me by agreeing to supervise this research work. I am sincerely grateful for their valuable advice, their scientific support and their availability.

I would also like to express my sincere thanks to Professor Farouk YALAOU, Director of the Industrial Systems Optimization Laboratory (LOSI), to have me welcomed to his team. I thank him again for his continuous follow-up, for his constructive advice and for the quality of the ideas proposed during the exchanges which have been very useful for my research work.

I greet my doctoral friends, especially Oussama MASMOUDI, Abbas HAMZE, Nacef TAZI, Mourad BENTALEB, Ali RIDA, ..., for their collaboration. I am grateful to thank all Iranian students in Troyes for our good time after scientific works, exchanges of knowledge, skills during my study which helped enrich the experience.

I gratefully acknowledge the Grand-Est region in France and the European Development Fund regional (ERDF) for their financial supports during this thesis.

Last but not least, I would like to thank my family and my best friends for their support and encouragement during the time far away from home. I must acknowledge my dear parents, Abbas and Mehri, and my lovely wife, Elnaz. It is obvious that without their supports, my success would not have been possible.

Outline

List of Tables

List of Figures

List of Abbreviations

TOU	**T**ime **O**f **U**se
TEC	**T**otal **E**nergy **C**ost
GA	**G**enetic **A**lgorithm
MA	**M**emetic **A**lgorithm
HA	**H**euristic **A**lgorithm
EAS	**E**nergy **A**ware **S**cheduling
CPP	**C**ritical **P**eak **P**ricing
CPU	**C**om**PU**tational time
RTP	**R**eal **T**ime **P**ricing
NPE	**N**on **P**rocessing state **E**nergy
PE	**P**rocessing state **E**nergy
GHG	**G**reen**H**ouse **G**as
NSGA-II	**N**on-dominated **S**orting-based **G**enetic **A**lgorithm II
DP	**D**ynamic **P**rogramming approach
Ton	**T**urn **on**
Toff	**T**urn **off**
Obj	**Obj**ective value
pmtn	pree**m**pti**o**n
LB	**L**ower **B**ound
cons	**cons**traint
var	**var**iable

List of Symbols

n	Number of jobs
N	Total number of jobs when the machine has different speeds
T	Number of periods
s	State of the machine
d_s	The required duration for state s
β_1	Number of required periods for Ton
β_2	Number of required periods for Toff
e_s	The energy consumption of state s
φ_s	Set of periods number in which the machine is in state s
φ_s^*	Optimal set of periods number in which the machine is in state s
c_t	Unit of energy cost in period t
p_j	Processing time of job j
$p_{j,i}$	Processing time of job j with speed i
P	Sum of the processing time
q_j	Energy consumption of job j
$q_{j,i}$	Energy consumption of job j with speed i
x	Number of extra periods
v_j	Number of possible speeds for job j
λ	Number of Ton/Toff states
Z	Objective value of any feasible solution
Z^*	Objective value of optimal solution
$x_{j,t}$	Job j is processed during period t
$y_{j,t}$	Job j begins to be processed in period t
$x_{j,i}$	Job j processed with speed i
$y_{j,i,t}$	Job j processed with speed i in period t
$R_{j,t}$	Sum of the energy unit costs if the machine starts to perform Job j in period t
C	Set of the energy cost for all the periods
$\tilde{C}$	Set of the energy cost for all the periods in increasing order
$\underline{C}_j$	The minimum cost of performing job j
H_l	Set of the possible nodes for level l
τ_k	Set of the possible levels for node k
$l_{min(k)}$	Earliest possible level for node k
$l_{max(k)}$	Latest possible level for node k
$Ev_{(k,l)-(k+1,l')}$	Edge's value for connecting node k in level l to node $k+1$ in level l'
$C_{(k,l)}$	Minimum cost for arriving to node k in level l
$A_{(k,l)}$	Set of the connected nodes to node k in level l
$\|V\|$	Number of vertices for the graph
$\|E\|$	Number of edges for the graph

Part I

English Version

Introduction

Outline of the current chapter

Background and relevance

In the last few years, economic and societal developments have led to a rapid increase in energy consumption and energy shortage has become a bottleneck to economic growth in many countries ([1]). Meanwhile, CO2 emissions generated from energy use became one of the main factors for global climate change and the greenhouse effect. Since the industrial revolution, the rate of greenhouse gases has increased up to 70% (between 1970 and 2004). The industrial sector is the largest consumer of energy in most of the countries, so, it is important to focus on the industrial sector to address the energy consumption minimization and the reduction of greenhouse gas emissions.

Moreover, since the turn of the millennium, in almost every industrial nation, electricity prices, one of the main energy sources used in the manufacturing factories, have been rising continuously (Fig. 1 and Fig. 2). This is mainly a result of taxes and duties to support the integration of renewable energies and the turning away from low-cost electricity generated by nuclear power. As a result, the share of the energy costs relative to the production costs increases, resulting in a lower competitiveness compared to countries with lower or more slowly increasing electricity prices.

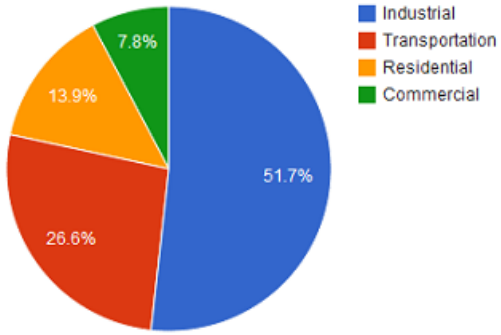

Figure 1 – World Energy Consumption by sector, 2012 (based on U.S. EIA Data)

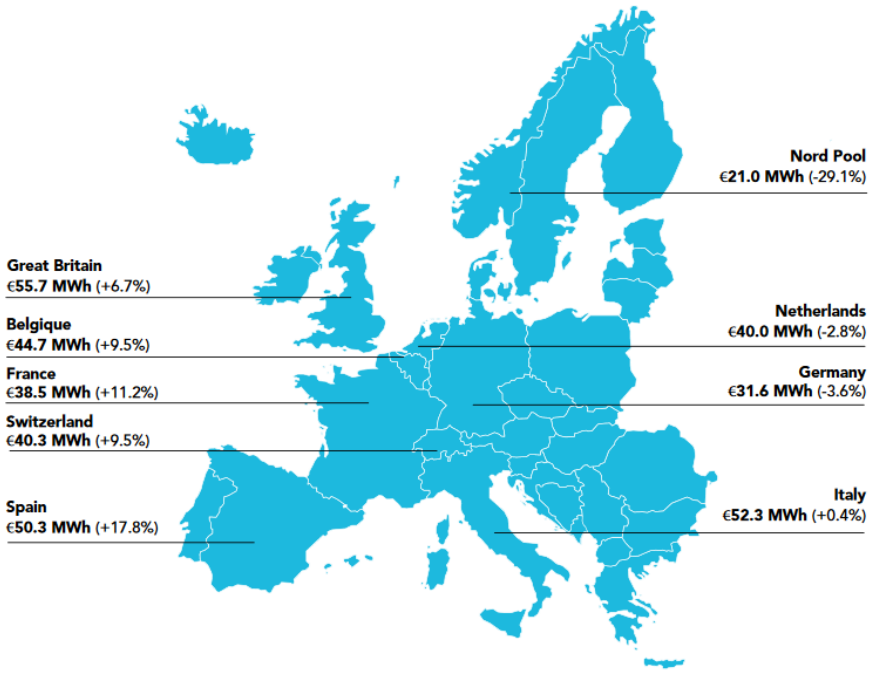

Figure 2 – European average electricity prices exchanges in 2015 and change vs 2014 (annual electricity report 2015 (Rte))

Due to increasing electricity price fluctuations, savings in energy costs without capital-intensive investments are possible by implementing specific organizational methods. These methods attempt to process energy-intensive orders at times of low prices and energy-low orders at times of high prices. All these issues have encouraged many researchers all around the world to study how to

improve the efficiency of a production system in terms of electricity consumption perspective to reduce production costs and environmental impact.

In the literature, the studies dealing with the energy efficiency of manufacturing systems can be divided in three different categories: papers which address energy consumption value minimization, papers on energy consumption cost or operational cost reduction, and problems with energy constraints.

The energy consumption minimization has been conducted in different perspectives such machine-level, product-level and system-level. Machine-level studies focus on developing and designing more efficient machines and equipment. Researchers concentrate on product design to reduce energy consumption in the product-level. Both the machine redesign and product redesign need enormous financial investments. In the system-level perspectives, manufacturers can achieve a significant reduction in energy consumption with fewer investment costs (by using decision models and optimization techniques to apply at production planning and scheduling decision levels). In this thesis, the system-level perspective is addressed to present some decreasing energy consumption methods for our problem.

The total energy consumptions of a production system consists of the amount of energy consumed during the non-processing states (NPE) (start-up, transition between different states, shut down and idle states), and during the processing state (PE). Besides, the amount of energy consumption in each system depends on the machine, the machine state, the processing speed, and the processed job. Therefore, decision makers may focus on the NPE and/or PE parts of any system to reduce its energy consumption. For this purpose, one of the most popular approach is investigating the NPE and PE consumption and using a scheduling method to change the processing jobs or machines order, and the machine's state or speed during a time period by shifting from on-peak periods to off-peak ones.

A comprehensive review of previous studies on production scheduling and energy consumption problem shows that there are different methods to integrate energy consumption concept in

# Problem	Single machine	Multi-states	Unique energy for the jobs	Different energy for the jobs	Uniform-speed	Speed-scalable
Pb1	✖	✖	✖		✖	
Pb2	✖	✖		✖	✖	
Pb3	✖	✖		✖		✖

Figure 3 – Three main sets of the studied problems in this thesis

production scheduling problem. Merkert et al:[2] introduced several options on how enterprise-wide optimization concepts can integrate energy management and scheduling. Gahm et al:[3] investigated scheduling approaches which aims to improve energy efficiency. They classified the literature based on three aspects which consist of energetic coverage, energy supply and energy demand. Finally, they developed a framework for energy efficient scheduling.

Contributions

In this thesis, we focus on the energy consumption minimization of a single machine system. To the best of our knowledge, few publications exist in the literature addressing the energy efficiency of a multi-state single machine system with the time-dependent electricity cost. For this purpose, a single machine system whom energy consumption depends on the machine's state is considered and several problems with different assumptions are studied. Three main variants of this problem are studied in this thesis. They are respectively: multi-states single machine scheduling problem with the same energy consumption for the jobs, multi-states and uniform speed single machine scheduling problem with different energy consumptions of the jobs, and multi-states and multi-speeds single machine scheduling problem with different energy consumptions of the jobs. These problems and their assumptions are presented in figure 3. The complexity of each problem and some sub-problems are analyzed and several solution methods are presented.

Outline

This thesis is organized into five chapters. In chapter 1, a brief definition about the different kinds of scheduling problems as well as a literature review of the studies which deal with production

scheduling and energy consumption problems are provided. The description is divided into several sections, including the studies which consider the environmental impacts of the energy consumption such as carbon emission, the papers that study decreasing energy consumption value, the work which try to reduce energy consumption cost or operational cost and the problem with energy sources constraint. Existent works and results are analyzed and the research gap is identified.

In chapter 2, the basic problem which deals with a multi-states single machine scheduling problem under time-of-use electricity tariffs, is introduced. A new mathematical model for the fixed sequence case is presented. Computational experiments on randomly generated instances are conducted to evaluate the proposed model and compare to an existing model. The complexity of this problem is analyzed by using a dynamic programming approach.

Chapter 3, presents a mathematical model for the problem which considers production scheduling at the machine-level and job-level simultaneously, to acquire minimum energy consumption cost. It means that in this case, the sequence of the jobs is not fix. So, the model searches firstly an optimal schedule for the state of the machine in each period and secondly it searches an optimal schedule for the jobs during the processing states. The proposed model is solved by CPLEX software. This problem is proved to be NP-hard and a new heuristic algorithm and a genetic algorithm are proposed to obtain good solutions. A large number of randomly generated instances are used to evaluate the proposed model and the performance of the proposed algorithms. Moreover, the complexity of this problem under different cost function assumption are analyzed and some lower bounds are presented for the general problem.

Chapter 4, investigates the problems when the energy consumptions are also related to the processing jobs, in two cases: uniform speed case and speed scalable case. It studies the complexity of these problems in two versions (with and without fixed sequence for the jobs) and proposes the mathematical models and some solution methods for each one.

Finally, chapter 5 concludes this thesis. It summarizes the different contributions ans discusses the limitations of the present research. It also introduces some potentially promising directions for future research.

1

Energy-efficient scheduling problems: state of the art

Outline of the current chapter

1.1 Introduction

Since the third industrial revolution which used electronics and information technology to automate production (between 1970 and 2004), the rate of greenhouse gases has increased. It caused climate change problem that concerns individuals and societies. Therefore, environmental protection is a major and essential issue. Moreover, in most of the countries, the unit of electricity price, as one of the main energy sources of the manufacturing sector, is rising continuously. Besides, a significant part of production costs for most of the industry is related to the energy consumption costs. Consequently, the production costs for producer is growing, resulting in a lower competitiveness compared to countries with lower or more slowly increasing electricity prices ([4]). These issues have encouraged many researchers all around the world to improve the electricity consumption efficiency of a production system with the aim of reducing the production costs and the ecological aspects, simultaneously.

Many works exist in the literature which integrate the ecological aspect in decision problems. They can be categorized based on the considered decision level: strategic, tactical and operational. At the strategic level, usually the papers are related to supply chain problems in which researchers want to determine the location of the sites, the capacities and the number of distribution centers, in order to minimize total costs and emissions of greenhouse gases. For example, Rodoplu et al. in [5] addressed a new single item lot sizing problem, including different energy sources to determine the optimal production planning and energy contract which minimize production and energy costs with respect to constraints of production systems and energy supplier contract conditions. At the tactical level, the papers deal with optimization of planning and resources management to manage energy consumption to limit greenhouse gas emissions and waste. For example, Masmoudi et al. in [6] and [7] considered energy constraints and different energy costs during the planning horizon of a flow-shop system in a lot-sizing problem. At the operational level, the research studies address the scheduling problems to maximize the performance of a production line with considering the energy factors.

This thesis deals with the scheduling problem which is in the operational level. A compre-

hensive review of previous works shows that the energy consumption of a manufacturing system can be minimized at three levels: machine, product and system ([8]). From the machine-level perspective, researches attempt to concentrate on developing and designing more energy-efficient machines and equipment to reduce power and energy demands of machine components. From the product-level perspective, researchers effort focus on modelling embodied product energy framework from the product design point of view. Note that, at the machine and the product levels, enormous financial investment and times are needed to design the new machine(s) or product(s) which consume less energy than the previous one. The benefits could not be interesting for most of manufacturing companies, especially those small and medium sized enterprises. At the system-level, manufacturers may reduce the energy consumption of their system by using several existing decision models and optimization techniques to manage the production plan or the schedule of the related system.
The system-level considered in this thesis is addressed to reduce energy consumption of a production system. We concentrated to the production scheduling problems which address the energy consumptions.

Chapter 1 introduces a brief synthesis of the scheduling problems. Then, we focus on the energy efficient single machine scheduling problems and a comprehensive review of the previous studies is presented. The papers are categorized in four general sets, the studies which consider the environmental impact of energy consumption such as carbon emissions, the works which studied the amount of energy consumptions, the researches addressing the energy consumption costs minimization, and the papers with energy constraints.

1.2 Scheduling problems

1.2.1 Definition

Production scheduling is one of the most important activities of a company at the operational-level that cause to remain competitive in demanding consumer markets. The problems of production scheduling and sequencing refer to decision making regarding the designation of jobs to available

resources and their subsequent order to optimize pre-defined performance measures. The relevance and potential of research and application in this area is enormous for both manufacturing and service companies, which has led researchers to address the problems of production scheduling from various perspectives over the past decades ([9]).

Depending on the studied problems, the resources can be considered as the machines in an assembly plant, computational resources such as CPU, memory and I/O devices in a computer system, runways at an airport, mechanics (human) in an automobile repair shop, etc. Activities can be considered as different operations in a manufacturing system, execution of a computer program, landings and take-offs at an airport, car repairs in an automobile repair shop, and so on ([10]).

In this study, we focus on the existing scheduling problems in a manufacturing system to improve production efficiency and reduce the production costs.

1.2.2 Scheduling in manufacturing

In a manufacturing system, when the orders are released, they are considered as the jobs with associated due dates. These jobs often have to be processed in a sequence by the machines in a work-center. The scheduling problems arise in this context. In scheduling terminology, a distinction is often made between a sequence and a schedule. A sequence usually corresponds to the order in which the jobs are processed on a given machine, while, a schedule usually refers to an allocation of jobs within a more complicated setting of machines, indicating the sequence on each machine. Assuming that m machines $M_i(i = 1, ..., m)$ have to process n jobs $J_j(j = 1, ..., n)$. The schedules may be machine-oriented like Fig. 1.1. This means that the schedule is for each machine and it represents an allocation of one or more time intervals to one or more jobs. It also may be job-oriented as illustrated at Fig. 1.2. An allocation of one or more time intervals to one or more machines for each job is done.

In a production system, some unforeseen events on the shop floor, such as machine breakdowns or longer-than-expected processing times, may have a major impact on the schedules. Moreover, the shop floor is not the only part of the organization that impacts the scheduling process. For example, processing of the jobs may sometimes be delayed if certain machines are busy, and the

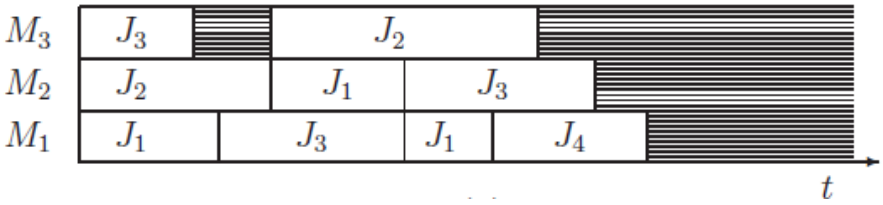

Figure 1.1 – Gantt chart for a machine-oriented schedule ([11])

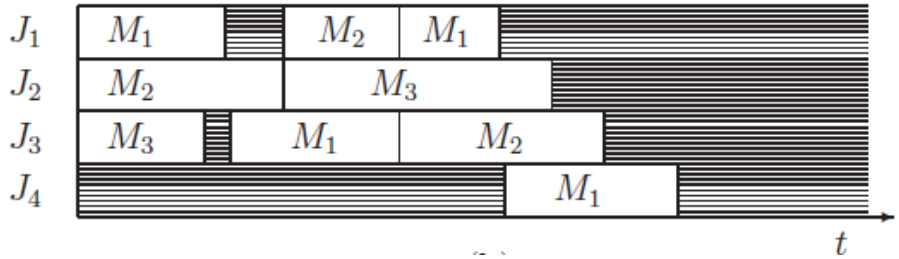

Figure 1.2 – Gantt chart for a job-oriented schedule ([11])

preemptions may occur when high priority jobs arrive while machines are busy. The decisions made at this higher planning level for the entire organization may also impact the scheduling process directly. In such an environment, the development of a detailed task schedule helps to maintain efficiency and control of operations. Therefore, the scheduling function has to interact with other decision-making functions that have to be taken into account. Fig. 1.3 illustrates the information flow in a manufacturing system. Regarding to the schedulers activities, 3 different classes of schedules may be defined as follows ([12]).

- **Non-delay schedule**: No machine is kept idle while a job is waiting for processing in a feasible schedule.

- **Active schedule**: No job can be put into an empty hole earlier in the non-preemptive schedule without loosing it's feasibility.

- **Semi-Active schedule**: No job can be completed earlier without changing the processing order on any machine in a feasible non-preemptive schedule.

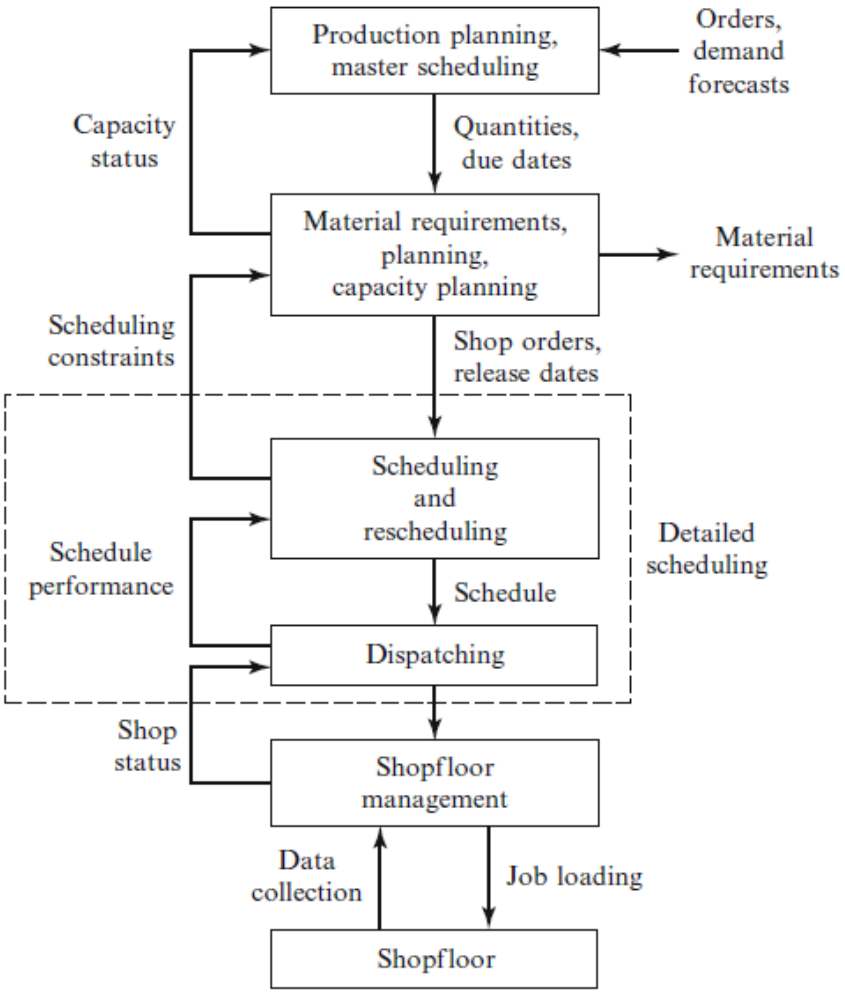

Figure 1.3 – Information flow diagram in a manufacturing system ([12])

1.2.3 Scheduling problems classification

In the literature, different definitions and classifications of the scheduling problems can be found. One of the most famous classification for classical scheduling problems is presented in [13]. The authors introduced the 3-field notation which is called Graham's notation $(\alpha|\beta|\gamma)$. In this notation, the first field (α) defines the machine environment. The second field (β) describes jobs' characteristics and scheduling constraints. It may contain multiple entries or no entry at all. The main possible symbols and their definitions for the fields α and β are listed in Table 1.1. Any other entry that may appear in the β field is self-explanatory. For example, $p_j = p$ implies that

α		β	
symbol	description	symbol	description
1	Single Machine	$pmtn$	Preemptions
Pm	Parallel and Identical Machines	nwt	No-Wait
Qm	Uniform Machines	$prec$	Precedence Constraints
Rm	Unrelated Machines	$batch(b)$	Batch processing
Jm	Job Shop	r_j	Release Dates
FJc	Flexible Job Shop	d_j	Due Date
Fm	Flow Shop	$\bar{d}_j$	Deadline
FFc	Flexible Flow Shop	w_j	Weigh
Om	Open Shop	nbr	Restrictions on the Number of Jobs
-	-	n_j	Restrictions on the Number of Operations for each Job

Table 1.1 – Description for the using symbols in Graham's notation: part1

γ	
symbol	description
$C_{max} = max(C_1, \cdots, C_n)$	Makespan
$TCT = \Sigma_{j=1}^{n} C_j$	Total Completion Time
$TWCT = \Sigma_{j=1}^{n} w_j.C_j$	Total Weighted Completion Time
$L_{max} = max(L_1, \cdots, L_n)$	Maximum Lateness
$TT = \Sigma_{j=1}^{n} T_j$	Total Tardiness
$TWT = \Sigma_{j=1}^{n} w_j.T_j$	Total Weighted Tardiness
$\Sigma_{j=1}^{n} U_j$	Total Number of Tardy Jobs
$\Sigma_{j=1}^{n} w_j.U_j$	Total Weighted Number of Tardy Jobs

Table 1.2 – Description for the using symbols in Graham's notation: part2

all processing times are equal, and $d_j = d$ implies that all due dates are equal. The third field (γ) in Graham's notation provides the objective function to optimize which usually contains a single entry ([10]). In scheduling problems, the objective to be minimized is often a function of the completion times of the jobs. Regarding to a schedule, let C_j be the completion time of job j. Based on the due date of each job (d_j), the lateness of job j is defined as $L_j = C_j - d_j$ and it's tardiness is defined as $T_j = max(L_j, 0)$. The difference between tardiness and lateness is that the tardiness is never negative. The unit penalty for a tardy job j is defined as $U_j = 1$ if $C_j > d_j$; otherwise, $U_j = 0$. The earliness of job j is defined as $E_j = max(d_j - C_j, 0)$. The lateness, the tardiness, the earliness, and the unit penalty are the four basic due date related penalty functions in the scheduling problems. Some possible symbols and their definitions for γ field are listed in Table 1.2.

Also, [11] proposed new classifications for new cases which are not included in [13]. The authors have taken into account the cases where machines are dedicated to performing certain types of tasks as well as the case where a machine can handle several tasks in parallel. Other properties of scheduling problems, like deterministic or stochastic problems, that follow laws of probability

Polynomial	Pseudo-polynomial	NP-hard
$1\|batch(b)\|C_{max}$	$1\|\|\Sigma w_jU_j$	$1\|r_j\|\Sigma C_j$
$1\|r_j,\ p_j=p,\ prec\|\Sigma C_j$	$1\|r_j,\ pmtn\|\Sigma w_jU_j$	$1\|prec\|\Sigma C_j$
$1\|r_j,\ pmtn\|\Sigma C_j$	$1\|batch(\infty)\|\Sigma w_jU_j$	$1\|r_j,\ pmtn,\ tree\|\Sigma C_j$
$1\|batch(b)\|\Sigma C_j$	$1\|\|\Sigma T_j$	$1\|r_j,\ pmtn\|\Sigma w_jC_j$
$1\|tree\|\Sigma w_jC_j$		$1\|r_j,\ p_j=1,\ tree\|\Sigma w_jC_j$
$1\|batch(\infty)\|\Sigma w_jC_j$		$1\|p_j=1,\ prec\|\Sigma w_jC_j$
$1\|prec\|L_{max}$		
$1\|r_j,\ pmtn,\ prec\|L_{max}$		$1\|r_j\|L_{max}$
$1\|batch(\infty)\|L_{max}$		$1\|batch(b)\|L_{max}$
$1\|\|\Sigma U_j$		$1\|r_j\|\Sigma U_j$
$1\|r_j,\ pmtn\|\Sigma U_j$		$1\|p_j=1,\ chains\|\Sigma U_j$
$1\|batch(\infty)\|\Sigma U_j$		$1\|batch(b)\|\Sigma U_j$
$1\|r_j,\ p_j=p\|\Sigma w_jU_j$		$1\|r_j\|\Sigma T_j$
$1\|r_j,\ p_j=p,\ pmtn\|\Sigma w_jU_j$		$1\|batch(b)\|\Sigma T_j$
$1\|r_j,\ p_j=1\|\Sigma w_jT_j$		$1\|\|\Sigma w_jT_j$

Table 1.3 – Complexity classification of single machine scheduling problems ([12])

for tasks and machines characteristics are also considered.

1.2.4 Scheduling problems complexity

To study scheduling problems, we also need to address their complexity such as [14] and [15]. Complexity theory is an important tool in scheduling research, which provides a mathematical framework for classifying problems as “easy” or “hard”, based on the problem solving times. Regarding to the complexity theory, the scheduling problems can be divided into 3 classes such as: polynomially solvable, pseudo-polynomially solvable, and NP-hard problems.

Table 1.3 presents a sample of complexity classification of fairly general single machine scheduling problems. For each scheduling problem the very first step is to try to propose a solution method for solving the problem. According to the complexity of each problem, different solution methods like exact and heuristic methods have been proposed. Some scheduling problems can be solved exactly by using standard techniques, such as branch-and-bound and dynamic programming methods or linear programming. These approaches have limits in terms of the size of the problem because of the computation time. Another possibility is to apply approximation algorithms to propose a good quality solution for the large size problems by reducing computing time. The heuristics and meta-heuristics algorithms like genetic algorithm are other approximation alternatives for solving the NP-hard problems. These methods cannot guaranteed the optimality of the solutions.

Depending on the types of problems and their complexities, for the instances larger than a specific size, it is not possible to obtain an exact solution in a reasonable computational time. That’s why

we use approximate methods. Also, we need to define some indicators to evaluate the performance of these methods. For this purpose, two most usual performance indicators in the literature are: the GAP (the difference between the obtained solution by the proposed method and the obtained solution by the exact method or a lower bound in percent), and it's execution time.

After a brief definition on the scheduling problems, in the following section, we give a comprehensive review of the studies which deal with energy concepts.

1.3 Energy-aware scheduling (EAS) problems

In the literature, there exist different approaches which integrate the energy concept in production scheduling problem. For example, [16] presented a literature review of decision support models for energy efficient production planning. [2] introduced several options on how enterprise-wide optimization concepts can integrate energy management and scheduling. [3] investigated scheduling approaches which aims to improve energy efficiency. They classified the literature based on three aspects: energetic coverage, energy supply and energy demand. They also developed a framework for energy efficient scheduling.
In this thesis, we divide the presented problems in the literature within different categories: the papers which consider the environmental impacts of the energy consumption such as carbon emission; the papers that study decreasing energy consumption value; the paper which try to reduce energy consumption cost or operational cost; and the problem with energy sources constraint.

1.3.1 Carbon emission

Regarding to the 5th assessment report of the Intergovernmental Panel on Climate Change in 2014, global annual greenhouse gas (GHG) emissions reached an all-time high of 49.5 billion tons carbon dioxide equivalent (GtCO2eq) in 2010. Of these 49.5 GtCO2eq/yr, industry directly contributed 18% of total emissions. Besides, industry indirectly contributed another 10.6% to emissions. In a result, 28.6% of global greenhouse gas emissions is contributed by industry sections. Therefore, the reduction of carbon emissions generated from energy consumption in production

systems is necessary and meaningful. For this purpose, in some cases, the authors deal with the carbon emission of the system.

For example, [17] considered the renewable energy as well as a related rechargeable battery and studied the minimization of total carbon emission which includes three terms: task production, procured power, equipment maintenance and daily operations. [18] dealt with minimizing of the total carbon dioxide emission and total completion time as a multi-objective optimization model. They focused on a single machine system with deterministic product arrival times and they emphasized the machine's energy consumption during idle and switched states. [19] examined carbon footprint in the production scheduling problem. They studied two multi-objective problem of a batch-processing machine followed by two parallel-processing machines to minimize the total weighted tardiness, carbon footprint and peak power. [20] presented a mathematical programming formulation to minimize total weighted flow time and carbon dioxide emission by considering renewable energy uncertainty, rechargeable battery and a weight for the importance of jobs. [21] developed the ϵ-archived genetic algorithm (ϵ-AGA) to examine two batch scheduling problems with the goal of minimising CO_2 emissions and the traditional due date-based objective of minimizing total weighted tardiness (TWT). [22] presented a new mathematical programming model of the flow shop scheduling problem that considers peak power load, energy consumption, and associated carbon footprint in addition to cycle time. In addition to the processing order of the jobs, the proposed scheduling problem considers the operation speed as an independent variable, which can be changed to affect the peak load and energy consumption. [23] investigated a scheduling problem in dual-resource constrained (DRC) job shop with interval processing time and heterogeneous resources. A lexicographical method is applied to minimize Interval carbon footprint and makespan.

1.3.2 Energy consumption

In the studies considering energy consumption, the decision maker has to decide the timing and the length of OFF/ON operations in addition to optimize the sequence of jobs. There are different ways to analyze the energy consumption of a manufacturing system. For example, [24] proposed a new parallel bi-objective hybrid genetic algorithm by considering makespan

and energy consumption for a scheduling precedence-constrained parallel applications problem. A multi-objective model that minimizes the energy consumption and total completion time is proposed in [25] to develop the operational methods by using dispatching rules.

For the systems with more than one machine, in most cases, the energy consumption depends on the machine (machine-dependent). Moreover, several states are considered for each production system, which can divided in two main sets: processing and non-processing states. The system's energy consumption is usually composed of the amount of consumed energy during the non-processing states (NPE) (start-up, transition between different states, shut down and idle states), and the amount of consumed energy during the processing state (PE). Usually, within these different states, the machine consumes different amount of energy (state-dependent). Moreover, several characteristics may change the energy consumption of the machine during the processing states such as: type of the processed job (job-dependent) and the processing speed of the machine (speed-dependent). In the following, a descriptions of the papers which studied each of these characteristics are presented.

Machine-dependent

Within the papers which studied the machine-dependent energy consumption, [26] proposed a modeling method of task-oriented energy consumption for machining manufacturing system. The energy consumption characteristics driven by task flow in this system are analyzed, which describes that energy consumption dynamically depends on the flexibility and variability of task flow in production processes. [27] introduced a uniform parallel machine scheduling problem where the objective is to minimize resource consumption such that the maximum completion time does not exceed a certain level. They shown that the problem is strongly NP-hard. [28] studied a multi-objective Flexible Job-shop Scheduling Problem (FJSP) optimization model, in which the makespan, processing cost, energy consumption and cost-weighted processing quality are considered. [29] focused on classical job shop environment which is widely used in the manufacturing industry. A model for the bi-objectives problem that minimizes total electricity consumption and total weighted tardiness is developed and a Non-dominant Sorting Genetic Algorithm is employed to obtain the Pareto front. [30] investigated an energy-efficient permutation flow shop scheduling problem with sequence-dependent setup and controllable transportation

time from a real-world manufacturing enterprise. A novel multi-objective mathematical model considering both makespan and energy consumption is formulated in this paper. [31] analyzed the trade-off between minimizing makespan, a measure of service level and total energy consumption, an indicator for environmental sustainability of a two-machine sequence dependent permutation flow shop. [32] developed constructive heuristics and multi-objective genetic algorithms (MOGA) for a two-machine sequence-dependent permutation flow shop problem to address the trade-off between energy consumption and makespan. They leveraged the variable speed of operations to develop energy-efficient schedules that minimize total energy consumption and makespan.

State-dependent

Within the papers which studied the state-dependent energy consumption, [33] divided the electricity consumption of a machine into the non-processing electricity (NPE) and processing electricity consumption (PE). The NPE is associated to the consumed energy by the machine during start-up, shut-down and idle states. As indicated by [25], in many manufacturing companies, on average, machines stay idle 16% of the time during the production shift. So, manufacturers could reduce their energy consumption easily considering the NPE consumption. Scheduling the machine and the jobs, as well as, Turn off/ Turn on the machine to reduce the NPE are some typical electricity saving methods which can be applied to any type of manufacturing system ([19]). A scheduling method can reduce NPE consumption by changing the jobs' processing order and the machine's state during a production shift.

[34] presented a mathematical model to minimize energy consumption and total completion time of a single machine system with deterministic job arrival and service time, by turning off the machine instead of leaving it idle. For this purpose, they used a multi-objective genetic algorithm and dominance rules. [35] presented a framework to minimize the total energy consumption and the total tardiness of a single machine optimization problem with unequal release dates, simultaneously. In this paper, they considered idle and setup energy. [36] considered a single-machine scheduling problem with power-down mechanism to minimize both total energy consumption and maximum tardiness. The aim is to find an optimal processing sequence of jobs and determine if the machine execute a power-down operation between two consecutive jobs.

[37] investigated energy consumption reduction in production systems through effective scheduling

of machine startup and shutdown. [38] proposed an energy-efficient model for flexible flow shop scheduling (FFS). An improved, genetic simulated annealing algorithm is adopted to make a significant trade-off between the makespan and the total energy consumption to implement a feasible scheduling. [39] proposed an energy-saving optimization method that considers machine tool selection and operation sequence for flexible job shops. A mathematical model is formulated using mixed integer programming and the energy consumption objective is combined with a classical objective (makespan). [40] considered an unrelated parallel machine scheduling problem with energy consumption and total tardiness. This problem is compounded by two challenges: differences of unrelated parallel machines energy consumption and interaction between job assignments and machine state operations. [41] established a mixed-integer nonlinear programming model for hybrid flow shop scheduling problem with minimizing the energy consumption.

Job-dependent

For the papers that considered the job-dependent energy consumption, [42], and [43] deal with a scheduling problem with a cumulative continuous resource and energy constraints, where each task requires a continuously-divisible resource. The instantaneous resource usage of any task was limited by a minimum and maximum resource requirement. They presented a Mixed Integer Linear Program (MILP) based on an event-based formulation to minimize the amount of energy consumption and a hybrid branch-and-bound method is proposed to solve the problem.

Speed-dependent

The third factor which can change the energy consumption of a machine is it's processing speed, which is included in some studies. For example, [44], and [45] studied the complexity of a deadline-based scheduling problem under a variable processing speed for the preemptive and non-preemptive cases, with the aim of finding a feasible schedule that minimizes the energy consumption. [46] addressed the scheduling problem of a set of jobs characterized by their release date, deadline and processing volume on a single (or a set of) speed-scalable processor(s). Their goal was to find a schedule respecting the release dates and the deadlines of the jobs so that the total energy consumption to be minimized.

[47] modeled an energy-aware multi-organization scheduling problem. They considered energy as

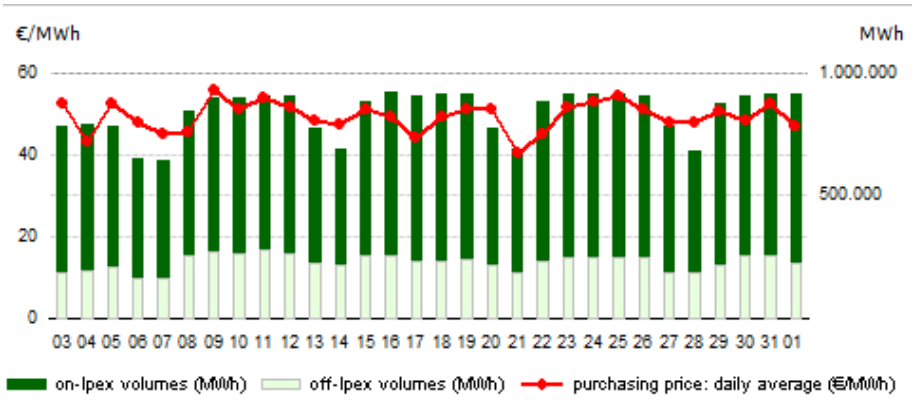

Figure 1.4 – Daily electricity price from 03/01/2018 to 01/02/2018 in Italy (GME,2018)

a resource, where the objective is to optimize the total energy consumption without increasing the energy spent by a selfish organization. They proved that the clairvoyant problem with variable speed processors and jobs with release dates and deadlines is NP-hard. [48] studied a flow-shop scheduling problem consisting of a series of processing stages and one final quality check stage with the aim of minimizing energy consumption. They consider that the product quality depends on its processing time at each stage, and the energy consumption is related to the processing speed, equipment state and product quality.

1.3.3 Energy cost

Another factor which may impact the total energy costs of any production system is variation of electricity prices during the time slots. For example, the evaluation of the electricity price in Italy during a working day and during 30 days are presented respectively in Fig. 1.5 and Fig. 1.4. In practice, electricity suppliers in different countries propose variable prices to balance the electricity supply and demand, to improve the reliability and efficiency of electrical power grids. In the literature there exist some papers which assumed the time-dependent energy cost to compute the total energy consumption cost. They reduced the total energy cost by shifting the on-peak hour energy consumption to the off-peak hour. The most common categories of time-varying rates are: Time-Of-Use (TOU), Critical Peak Pricing (CPP) and Real Time Pricing (RTP) ([49]).

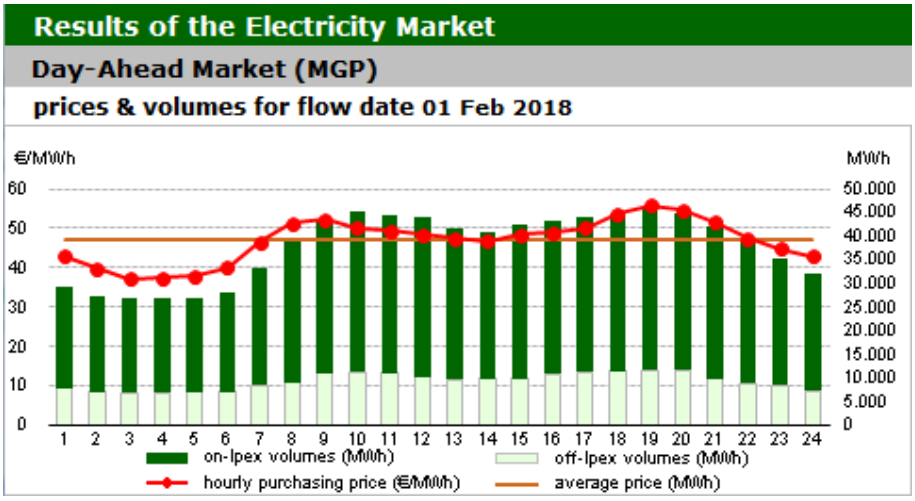

Figure 1.5 – Hourly electricity price for 01/02/2018 in Italy (GME,2018)

Time-Of-Use (TOU) A static TOU rate divides the day into time periods, and it provides a schedule of rates for each period. In this option, two types of periods are generally defined: "on-peak" and "off-peak". In some cases, TOU rates may have a shoulder (or mid-peak) period, or even two peak periods. The kWh energy charge during on-peak periods can be evidently higher than during off-peak periods, such as more than twice. In this rate, there is a certainty about what the rates will be and when they will occur. For example, in Fig. 1.6, the TOU tariff was taken from the electricity bill of a Belgian manufacturer. As shown in this figure, this price structure has two levels where the off-peak period lasts from 9 PM to 6 AM the next day.

TOU rates encourage permanent load shifting away from peak hours. It also could be used to encourage adoption of plug-in electric vehicles, solar photovoltaic systems, and distributed energy storage technologies by providing lower rates during the optimal time of charging (off-peak) and higher rates during the time of discharge or selling back to the grid. TOU rates are not dynamic, since they are not dispatched based on the changes in actual wholesale market prices or in reliability-related conditions. They are therefore less useful for addressing specific events on the grid and integrating variable renewable energy resources. Consequently, TOU rates do not provide as large a peak load reduction as dynamic rate designs due to the price signal being averaged over a large number of peak hours instead of a relatively limited number of very high-priced hours.

Within the papers which studied the impact of TOU tarrifs in some manufacturing systems, for

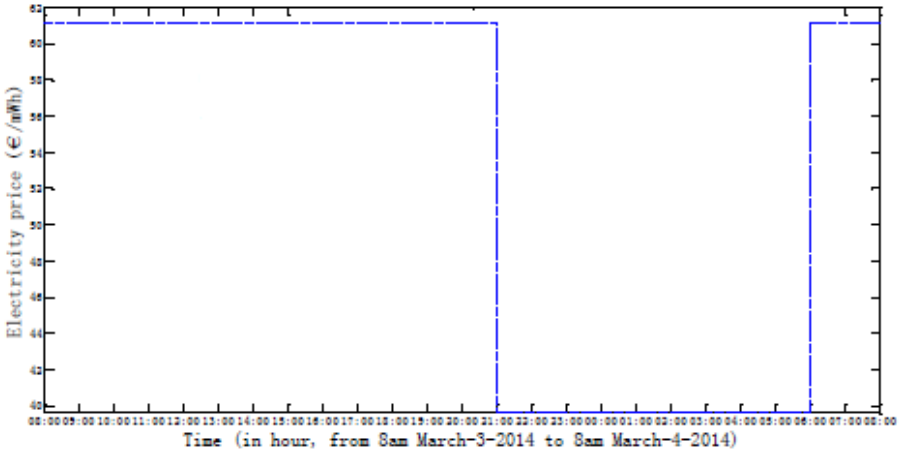

Figure 1.6 – Time-Of-Use electricity tariff ([50])

example, Ding et al. [51] addressed the unrelated parallel machine scheduling problem under a TOU pricing scheme. Their objective was to minimize the total electricity cost by appropriately scheduling the jobs such that the overall completion time does not exceed a predetermined production deadline. They presented a new time-interval-based mixed integer linear programming formulation. Moreover, they reformulated the problem using Dantzig–Wolfe decomposition and proposed a column generation heuristic to solve it. [52] deals with a typical batch-type and energy intensive process in steel industry. A multi-objective production scheduling optimization model is proposed under TOU pricing, to minimize the power costs on the premise of ensuring the product quality. A NSGA-II based production scheduling algorithm is also proposed to generate Pareto-optimal solutions. [53] studied the steel making-refining continuous casting scheduling problem with considering variable electricity price to reduce the electricity cost and the associate production cost. They proposed a decomposition approach for this problem.

Critical Peak Pricing (CPP) Under a CPP rate, participating customers pay higher prices during the few days when wholesale prices are the highest or when the power grid is severely stressed. This higher peak price reflects both energy and capacity costs. As a result, the portion of those costs is spread over relatively few hours of the year. In return, the participants receive a discount on the standard tariff price during the other hours of the season or year to keep the total annual revenue constant. Customers are typically notified of an upcoming “critical peak

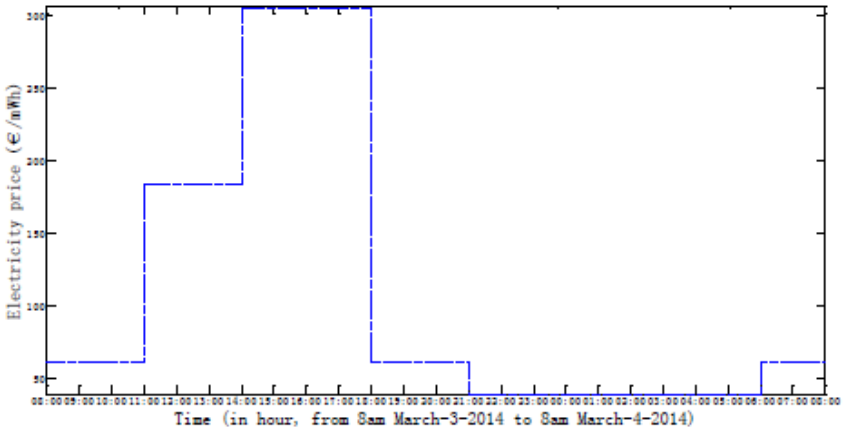

Figure 1.7 – Critical Peak Pricing electricity tariff ([50])

event" one day in advance. For example as presented in Fig. 1.7, the moderate price lasts from 11 AM to 2 PM of 3 March, and the high price has a period from 2 PM to 6 PM on the same day. Like the TOU rate, the CPP rate is simple for customers to understand. It provides a strong price signal and has produced some of the highest observed peak reductions among participants. In addition, it exposes customers to higher prices during only a very limited number of hours. But, political acceptance of the rate is sometimes limited due to the relatively high critical peak price. Furthermore, some customers consider the CPP rate to be more intrusive than a TOU rate because customers are contacted each time a critical event is called.

As the works which studied CPP rate, for example, [54] addressed the scheduling of continuous single stage multi-product plants with parallel units and shared storage tanks where the processing tasks are energy intensive. They considered time-dependent electricity pricing and availability together with multiple intermediate due dates. A new discrete-time aggregate formulation is proposed to plan the production levels quickly. [55] studied an off-line scheduling problem arising in demand response management in a smart grid. The electricity cost is measured by a convex function of the load in each time slot. Their objective was to schedule all requests with the minimum total electricity cost.

Real Time Pricing (RTP) Participants in RTP programs pay for energy at a rate that is linked to the hourly market price for electricity. Depending on customer class, participants are

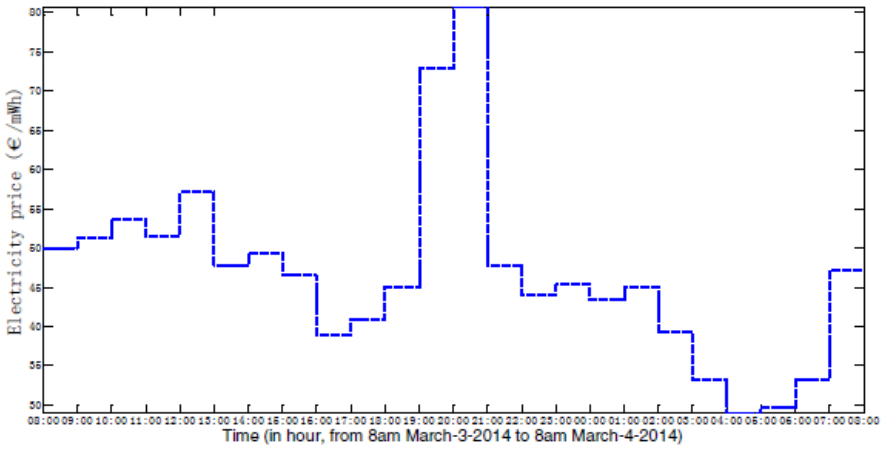

Figure 1.8 – Real Time Pricing electricity tariff ([50])

made aware of hourly prices on either a day-ahead or hour-ahead basis. The main advantage of RTP rates is that they provide the most granularity in conveying accurate hourly price signals to customers. These rates also provide a dynamic price signal that responds to modifications in market conditions. Generally, without automation technologies it is difficult for customers to respond to an hourly basis prices.

Fig. 1.8 illustrates the RTP data which is taken from the Belgian electricity spot market, where the hourly dynamic electricity price is known one day in advance.

In the literature, as a work which studied RTP rate, [56] developed a time-indexed integer programming approach to optimize the manufacturing schedule of a factory for minimal energy cost under real-time pricing (RTP) of electricity. [57] proposed a method for energy efficient and labor-aware production scheduling at the unit process level under real-time electricity pricing.

1.3.4 Energy consumption cost

In some other papers the effects of other factors such as machine-dependent, speed-dependent or state-dependent, in addition to time-dependent are investigated simultaneously. For example, [8] assumed the TOU electricity cost in a hybrid flow shop scheduling problem by considering the machine-dependent factor to minimize the makespan. [58], and [59] deal with the production and energy efficiency of the unrelated parallel machine and flexible job-shop scheduling problem. They minimized total tardiness and total energy consumption cost by considering the time-dependent

and machine-dependent electricity costs.

[60] investigated online and off-line energy-efficient algorithms for flow time and total cost minimization when the machine has variable speed and energy consumption. [1] considered the scheduling problem of processing jobs with arbitrary power demands that must be processed at a single uniform speed or speed-scalable machine to minimize total electricity cost under a time of use electricity tariffs. They analyzed the complexity of these two problems in preemptive and non-preemptive cases. [61] proposed a mathematical model to minimize energy consumption costs for single machine scheduling by considering variable energy prices during a production shift and unique energy consumption for each state of the machine. For this purpose, they made decisions at machine level to determine the state of the machine at each period as well as specifying the sequence of the jobs in the process states of the machine.

[62] represented a new scheduling model to minimize the total time slot costs where operational costs of a job vary over time. [63] deal with the off-line demand scheduling problem with different power demand at each period. They supposed preemptively and non-preemptively demands scheduling. [64] assumed an energy-conscious single machine scheduling problem, when each processing job has its power consumption and electricity prices may vary from hour to hour throughout a day. The objective was to minimize the total electricity cost required for processing the jobs.

[65] considered scheduling problem of parallel machine when each machine has a cost per unit of time and the cost is the sum of the energy cost and clean up cost. [50] and [66] proposed a generic mixed-integer programming model for a single machine scheduling problem to minimize total energy costs at volatile energy prices without exceeding the due date. [67] investigated a preemptive scheduling problem with an energy constraint in each period and different energy consumption for each job. They assumed the electricity time-varying prices to minimize the total electricity consumption and operations postponement penalties costs.

[68] addressed a novel production scheduling method to minimize the energy cost when finite state machine, multiple processes idle modes and time varied electricity price are considered. [69] presented a bi-objective optimization of a single machine batch scheduling problem to minimize the makespan and total energy cost. They assumed the Time-of-use tariffs for electricity, and different energy consumption levels depending on the machine's temperature level.

As can be conclude from the presented literature review in this section, since TOU rates have a simple design that is predictable and easy for customers to understand, they are studied more than the others by researchers.

1.3.5 Energy constraint

Finally, in the last set of researches which are investigated in this study, the authors addressed production scheduling involving energy constraints. In these studies, the authors considered a capacity constraint of total electricity (energy) which is consumed by the machine in each period or in total. For example, the scheduling problem with continuous resource and energy constraint is addressed in [70]. The authors proposed a strongly polynomial extension of the standard energetic reasoning scheme for their problem. [71] addressed a scheduling problem with continuous resources and energy constraints. Given a set of non-preemptive activities, each activity requires a continuously divisible resource whose instantaneous usage is limited in maximum and minimum, its processing satisfying a time window and a total energy requirement. [72] presented the scheduling of continuous single stage multi product plants with parallel units and shared storage tanks. They considered time-dependent electricity pricing and availability together with multiple intermediate due dates, handled as hard constraints. [73] studied the schedule of the jobs in flexible flow shops in order to account power's peak. They presented an approach to modify the original timetable in order to reduce the shop floor power's peak while accepting possible worsening of the scheduling objectives (tardiness and makespan).

[74] considered various problems of scheduling under resource constraints on the system which place the restriction that not all machines can be run at once. These can be power, energy, or makespan constraints on the system. In the setting where there is a constraint on power, they showed that the problem of minimizing makespan for a set of divisible jobs is NP-hard. The scheduling to minimize energy with power constraints is also NP-hard. The scheduling with energy and makespan constraints with divisible jobs can be solved in polynomial time, and the problems with non-divisible jobs are NP-hard.

[75] considered a flow shop scheduling problem with a restriction on peak power consumption, in

addition to the traditional time-based objectives.

To make the difference between the studied works in this thesis, figures 1.9 to 1.12 mark the considered energy concept in each papers.

Energy aspects / References	Carbon emission	Energy cost	Energy constraint	Energy consumption			
				Machine	Job	State	Speed
Albers and Fujiwara (2007)							×
Antoniadis and Huang (2013)							×
Antoniadis et al. (2015)							×
Ramos and Leal (2017)		×		×			
Artigues and Lopez (2015)			×		×		
Bampis et al. (2015)							×
Bruzzone et al. (2012)			×	×	×		
Burcea et al. (2013)					×		
Castro et al. (2009)		×	×	×			
Castro et al. (2011)			×	×			
Liu et al. (2014)	×					×	
Che et al. (2017)						×	
Che et al. (2016)		×			×		
Che et al. (2017)		×		×	×		
Chen et al. (2013)				×		×	
Cheng et al. (2016)		×					
Cheng et al. (2017)		×				×	
Cheng et al. (2014)		×					
Cheng et al. (2016)		×					
Liu and Huang (2014)	×					×	
Liu (2015)	×		×			×	
Liu (2014)	×					×	
Cohen et al. (2014)				×			×
Total=23	**4**	**8**	**5**	**7**	**5**	**7**	**5**

Figure 1.9 – Literature review analysis: part 1

Energy aspects / References	Carbon emission	Energy cost	Energy constraint	Energy consumption			
				Machine	Job	State	Speed
Dai et al. (2013)				×		×	
Ding et al (2016)		×		×	×		
Dupty et al. (2016)			×			×	×
Giglio et al. (2017)				×		×	
Liu et al. (2016)						×	×
Gong et al. (2017)		×			×	×	
Gong et al. (2015)		×				×	
Gong et al. (2016)		×				×	
Hadera et al. (2016)					×		
Haït and Artigues (2009)			×		×		
Haït and Artigues (2011)			×		×		
Haouassi et al. (2016)			×		×		
He et al. (2012)				×	×	×	
He et al. (2015)				×	×	×	
Ji et al. (2013)				×			
Jiang et al. (2014)				×			
Moon et al. (2013)				×			
Fang et al. (2011)	×			×			×
Fang et al. (2013)			×	×			×
Fang et al. (2016)		×			×		×
Le and Pang (2013)				×	×		
Lee et al. (2017)						×	
Lei and Guo (2015)	×		×	×		×	
Total=23	**2**	**5**	**6**	**12**	**10**	**11**	**5**

Figure 1.10 – Literature review analysis: part 2

Energy aspects / References	Carbon emission	Energy cost	Energy constraint	Energy consumption			
				Machine	Job	State	Speed
Li et al. (2016)	×		×	×			
Liang et al. (2015)				×		×	
Liu et al. (2008)				×	×	×	
Liu et al. (2014)				×		×	
Lu et al. (2017)				×		×	×
Luo et al. (2013)		×		×		×	×
Mansouri et al. (2015)				×			
Mansouri and Aktas (2016)				×			×
Mashaei & Lennartson (2013)				×		×	
Masmoudi et al. (2015)		×	×	×			
Masmoudi et al. (2017)		×	×	×			
Mikhaylidi et al. (2015)		×	×		×		
Nattaf et al. (2016)			×		×		
Moon and Park (2014)		×		×			
Mouzon et al. (2007)						×	
Mouzon and Yildirim (2008)						×	
Nattaf et al. (2015)			×		×		
Ngueveu et al. (2016)			×		×		
Oron (2011)			×		×		
Pach et al. (2014)						×	
Plitsos et al. (2016)			×			×	
Rager et al. (2015)					×		
Różycki and Węglarz (2012)			×		×		×
Total=23	**1**	**5**	**10**	**12**	**8**	**10**	**4**

Figure 1.11 – Literature review analysis: part 3

Energy aspects / References	Carbon emission	Energy cost	Energy constraint	Energy consumption			
				Machine	Job	State	Speed
Shrouf et al. (2014)		×				×	
Tan and Liu (2014)		×	×	×	×		
Tan et al. (2015)		×			×		
Tang et al. (2016)					×	×	
Fang and Lin (2013)		×		×			×
Wang et al. (2011)	×				×	×	
Wang et al. (2016)		×				×	
Wang and Wang (2018)				×		×	×
Gong et al. (2016)		×				×	
He and Liu (2010)				×	×		
Yi et al. (2012)	×			×		×	
Yildirim and Mouzon (2012)						×	
Yin et al. (2016)	×			×		×	
Liu et al. (2014)				×		×	
Liu et al. (2015)		×		×		×	
Liu et al. (2016)				×		×	
Yusta et al. (2010)		×				×	
Zhang et al. (2012)				×			×
Zhang et al. (2017)		×		×		×	
Zheng and Wang (2015)	×		×		×	×	×
Li et al. (2016)				×		×	
Total=21	**4**	**9**	**2**	**12**	**6**	**17**	**4**

Figure 1.12 – Literature review analysis: part 4

1.4 Conclusion

Based on the presented literature review in this chapter, few works studied energy consumption and energy cost factors at the same time. Among these studies ,to the best of our knowledge, very few publications address the energy efficiency of a multi-states single machine system with the time-dependent electricity cost.

In this thesis, we are interested to study on this problem. For this purpose, a single machine system with three main states as ON (processing), Idle, and OFF, and two transition states as turning on and turning off are considered. Our aim is, at first, to analyze the complexity of this kind of problems, and finally propose some solving methods for them. Moreover, among the studied papers, we just found one article ([57]) which studied the total energy costs minimization in addition to the state-dependent and job-dependent energy consumption. Also, there exists just one paper ([76]) which studied the total energy costs minimization in addition to the state-dependent and state-dependent energy consumption, and there is no previous work considering state-dependent, job-dependent and speed-dependent energy consumptions all together. Therefore, several kind of this problem with different assumptions are studied in the following chapters.

2

Multi-states single-machine energy-efficient scheduling problem with fixed sequence

Outline of the current chapter

2.1 Introduction

This chapter deals with energy-efficiency of a multi-states single machine scheduling problem when the sequence of the jobs is fixed, and the TOU electricity tariffs are considered for each period. So, the energy costs are varying among the periods and the energy consumption of the machine's states are different. Therefore, in this problem the scheduling problem search an optimal schedule to allocate the states with low consumption in the peak time periods, and the states with high consumption in the off peak time periods.
This problem was introduced by Shrouf et al. [61]. They proposed a basic mathematical model for it.
In this chapter, we are attempted to propose an improved model for this problem first of all, and then, we present a new dynamic programming approach to model this problem and analyze its complexity.

The remainder of this chapter is organized into five sections. In section 2.2, the definition of the problem with its assumptions and constraints are presented. In section 2.3, the basic model for the considered problem is presented and an improved model is proposed. Moreover, the comparison between these two models is also represented. In section 2.4, the complexity of this problem is analyzed by using a new dynamic programming approach. Finally, the brief conclusion of this chapter is presented in section 2.5.

2.2 Problem presentation

This chapter addresses a scheduling problem of several jobs on a multi-states single machine within a given planning horizon. The horizon is divided into T periods with the same length which are characterized by their unit of energy price. Actually, depending on the availability of the energy sources and the demands at each moment, and some other parameters like the ecological taxes, the energy suppliers define a time-varying energy price for their customers. As it is described in the previous chapter, the most common categories of time-varying rates are Time-Of-Use (TOU), Critical Peak Pricing (CPP), and Real Time Pricing (RTP). In the whole of

this thesis, TOU is used to define the energy prices. So, in this study, c_t ; $\forall t = 1, ..., T$, indicates the unit of energy price at period t. The objective is to find the most economical production schedule for the machine's states in terms of total energy consumption costs (TEC), with making sure that all the necessary setups are done and the constraints between the states are respected.

2.2.1 Assumptions and constraints

The assumptions and the constraints of this problem are considered as follows.
The considered machine has three main states (ON, OFF, Idle) and two transition states. The transition states are for transiting from OFF state of the machine to ON state and vice versa (turning on and turning off procedures) named Ton and Toff, respectively.
It must be mentioned that, the transition time between Idle and ON states is neglected and the transition between Idle and OFF states is not allowed in our system. Each state of the machine is characterized by a specific energy consumption and a required periods number. That means, when the machine is in state $s \in \{OFF,\ Ton,\ ON,\ Toff,\ Idle\}$, it must remain in the same state during a fixed number of periods (d_s), whilst it consumes a specific amount of energy per period (e_s). So, the total energy consumption of the machine for each unit of time in state s is equal to $e_s \times d_s$.
The initial and final states of the machine are assumed as OFF states. In the other words, the machine must be in OFF state during initial ($t = 0$) and final ($t = T$) periods. Note that, in this study, period $t = 0$ is just for identifying the initial state of the machine which is OFF and the scheduling horizon is from period $t = 1$ to $t = T$.
The machine's energy consumption during OFF state is equal to zero ($e_{OFF} = 0$), and without loss of generality, the following relations are considered between the states' energy consumption:

$$e_{ON} > e_{Idle} > e_{OFF} = 0 \tag{2.1}$$

$$e_{Ton} > e_{OFF} = 0 \tag{2.2}$$

$$e_{Toff} > e_{OFF} = 0 \tag{2.3}$$

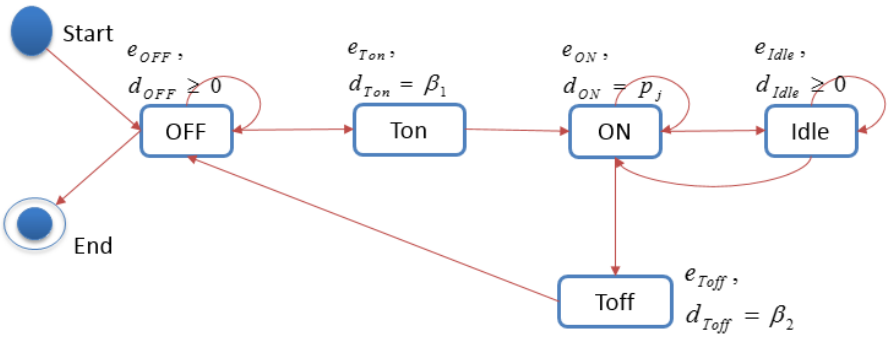

Figure 2.1 – Machine states and possible transitions.

All the jobs $j = 1, ..., n$ are available during whole the horizon (from period $t = 0$ to $t = T$) and they are characterized by their specific processing times p_j. The preemption is not allowed in this problem. It means that, if the machine goes in state s, it must be in the same state during d_s periods. So, for the processing jobs, if the machine was started to perform job j, it must continue to perform this job during p_j periods.

Moreover, generally the energy consumption during processing the jobs (ON states) depends on the performed job. Let define q_j the unit of energy consumption of the machine per unit of time when it processes job j. In the other words $e_{ON} = q_j$. For the presented problem in this chapter, it is considered that the energy consumption of the machine in ON state is constant and independent from the processed job. So, in this problem we have $e_{ON} = q$ and $q_j = q$; $\forall\ j = 1, ..., n$.

These assumptions and the possible transitions between different states are illustrated in Fig. 2.1.

2.2.2 Illustration

The predetermined order of the jobs is addressed for this problem. In the other words, the sequence of the jobs is fixed and it is required to just identify the optimal sequence for the machine's states.

As it is shown in Fig. 2.1, the first step is to choose the best period for turning on the machine. This transition takes β_1 periods and it consumes e_{Ton} units of energy per period. Then the machine is in ON state and is ready to process the first job $j = 1$, that takes p_1 periods and

Parameter	Value	Parameter	Value
β_1	2	β_2	1
p_1	3	p_2	2
p_3	4	p_4	2
p_5	3	e_{OFF}	0
e_{ON}	4	e_{Idle}	2
e_{Ton}	5	e_{Toff}	1

Table 2.1 – The parameters' values for an instance with 5 jobs and 32 periods

consumes $e_{ON} = q$ units of energy per period. Once the job is processed completely, there are three possibilities for the machine: it may stay in ON state and process the next job; it can go to Idle state for one period or more and also, the machine can go to OFF state. Note that the transition between ON and OFF states takes β_2 periods and it consumes e_{Toff} units of energy per period. Regarding to their total energy consumption costs, any of these possibilities may be selected. Moreover, when the machine is in Idle state, for the next period, it may stay in Idle state or pass to ON state.

The gant chart for an instance of 5 jobs and 32 periods with the parameters' values as Table. 2.1 is provided in Fig. 2.2. The unit of energy price for each period is presented at the second line of this figure.

In this solution, the machine is in OFF state until period 3, then it turns to the Ton state during periods 4 and 5. Since the sequence of the jobs is fixed and the preemption is not allowed, jobs 1,2 and 3 are processed from period 6 to 14, consequently. Since the price of energy are high in periods 15, 16, 17 and 18, the best solution is shutting down the machine. After turning on the machine during periods 19 and 20, it processed the jobs 4 and 5 from period 21 to 25. Finally, the machine goes to Toff state in period 26, and from period 27 to the end, the machine is in OFF states. Regarding to the unit of energy price in each period, the total energy consumption costs for this solution is equal to the 222.

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	Cost
c_t	0	8	8	8	4	4	4	3	3	3	2	2	2	2	2	10	10	10	10	3	3	3	2	2	2	2	6	6	3	3	3	5	5	
ON							16	12	12	12	8	8	8	8	8							12	8	8	8	8									
OFF	0	0	0	0													0	0	0									0	0	0	0	0	0	
Idle																																		
Turn on					20	20														15	15													
Turn off																10											6							
The schedule	Off				Ton		Job1				Job2	Job3				Toff	Off			Ton		Job4		Job5			Toff	Off						222

Figure 2.2 – An example for the general problem

2.3 Mathematical model

The primary step to deal with a scheduling problem is to provide a mathematical model for the problem and then find some solving methods for it. In the literature, the same problem was addressed by [61], where they proposed a basic model for it. In this section, first of all, the basic model ([61]) is presented and then an improved version of the model is proposed.

2.3.1 Shrouf et al.'s model

In the following, firstly the necessary sets, parameters and variables of the proposed model by Shrouf et al. [61] are introduced and then, the mathematical model including objective function and constraints are presented.

Parameters:

n: Number of jobs

T: Total number of periods

s: States of the machine (ON, OFF, Idle, Ton, Toff)

Note: to facilitate the modeling, the three possible states are considered as integer numbers (s=1 represents the ON state; s=2 for OFF state; s=3 for Idle state; s=21 for Ton state; s=12 for Toff state).

e_s: Amount of energy that machine consumes during state s

$e_{ss'}$: Amount of energy consumption when the machine is transiting from state s to state s'

$d_{ss'}$: Number of periods that must elapse when machine switches from state s to state s'

c_t: The energy cost in period t

p_j: Processing time of job j in number of periods

Decision variables:

$$\alpha_{s,t} = \begin{cases} 1 & ; \text{ If the machine is in state } s \text{ during period } t \quad (s = 1, 2, 3) \\ 0 & ; \text{ Otherwise} \end{cases}$$

$$\beta_{ss',t} = \begin{cases} 1 & ; \text{ If the machine is in transition from state } s \text{ to } s' \text{ in period } t \ (s, s' = 1, 2) \\ 0 & ; \text{ Otherwise} \end{cases}$$

$$x_{j,t} = \begin{cases} 1 & ; \text{ If job } j = 1, ..., n \text{ is processed during period } t \\ 0 & ; \text{ Otherwise} \end{cases}$$

$$y_{j,t} = \begin{cases} 1 & ; \text{ If job } j = 1, ..., n \text{ begins to be processed in period } t \\ 0 & ; \text{ Otherwise} \end{cases}$$

The problem was formulated as follows.

$$Pb_1 = Min \sum_{t=0}^{T} c_t \left(\sum_{s=1}^{3} e_s \cdot \alpha_{s,t} + \sum_{s=1}^{2} \sum_{s'=1}^{2} e_{ss'} \cdot \beta_{ss',t} \right) \tag{2.4}$$

$$\sum_{j=1}^{n} x_{j,t} = \alpha_{1,t} \qquad ; \forall\ t = 0, ..., T \tag{2.5}$$

$$\sum_{s=1}^{3} \alpha_{s,t} + \sum_{s=1}^{2} \sum_{s'=1}^{2} \beta_{ss',t} = 1 \qquad ; \forall\ t = 0, ..., T \tag{2.6}$$

$$\alpha_{s,t} \leq \sum_{s'=1 | d_{ss'}=0}^{3} \alpha_{s',t+1} + \sum_{s=1 | d_{ss''} \geq 1}^{2} \sum_{s''=1}^{2} \beta_{ss'',t+1} \qquad ; \forall\ t = 0, ..., T-1;\ s = 1, 2, 3 \tag{2.7}$$

$$\beta_{ss',t} \leq \beta_{ss',t+1} + \alpha_{s',t+1} \qquad ; \forall\ t = 0, ..., T-1;\ s, s' = 1, 2, 3;\ s \neq s';\ d_{ss'} \geq 1 \tag{2.8}$$

$$\sum_{p'=p+1}^{t+d_{ss'}} \beta_{ss',t'} \geq (\alpha_{s,t} + \beta_{ss',t+1} - 1) \cdot d_{ss'} \qquad ; \forall\ t = 0, ..., T-1;\ s, s' = 1, 2;\ s \neq s';\ d_{ss'} \geq 1 \tag{2.9}$$

$$\beta_{ss',t} + \beta_{ss',t+d_{ss'}} \leq 1 \qquad ; \forall\ t = 0, ..., T - d_{ss'};\ s, s' = 1, 2;\ s \neq s';\ d_{ss'} \geq 1 \tag{2.10}$$

$$\sum_{j=1}^{n} x_{j,t} \leq 1 \qquad ; \forall\ t = 0, ..., T \tag{2.11}$$

$$y_{j',t} \leq \sum_{t'=0}^{t} y_{j,t'} \qquad ;\forall\ t = 0, ..., T; \forall\ j, j' = 1, ..., n;\ \ j' > j \tag{2.12}$$

$$\sum_{t'=t}^{t+p_j-1} x_{j,t'} \geq p_j \cdot y_{j,t} \qquad ;\forall\ t = 0, ..., T;\ \ j = 1, ..., n \tag{2.13}$$

$$\sum_{t=0}^{T} x_{j,t} \leq p_j \qquad ;\forall\ j = 1, ..., n \tag{2.14}$$

$$\sum_{t=0}^{T} y_{j,t} = 1 \qquad ;\forall\ j = 1, ..., n \tag{2.15}$$

$$\alpha_{2,t} = 1 \qquad ;\ t = \{0, T\} \tag{2.16}$$

$$\alpha_{s,t},\ \beta_{ss',t} \in \{0, 1\} \tag{2.17}$$

$$x_{j,t},\ y_{j,t} \in \{0, 1\} \tag{2.18}$$

In this model, (2.4) deals with the objective function which computes the energy consumption costs depending on the machine status and the energy prices at each period. Equation (2.5) ensures that the machine is able to process a job, only when the machine is in processing state. Equation (2.6) expresses that, at each period, the machine must be in one of the states or in a transition between two states. Equations (2.7) and (2.8) limit the state of the machine at one period based on the status or transition that the machine has in the previous period. Equations (2.9) and (2.10) identify lower and upper number of periods in which the machine can be in a transition until the operations complete before end of the horizon time. Equation (2.11) specifies that the machine can process only one job during each period. Equation (2.12) ensures processing the jobs based on the given sequence. Equation (2.13) imposes the non-preemption of jobs. Equation (2.14) introduces the processing time of each job. Equation (2.15) forces all jobs to be processed during the time limitations. Equation (2.16) expresses the boundary conditions. In the next section, it is attempted to propose an improved mathematical model for this problem by keeping its assumptions and constraints.

2.3.2 Improved model

In the basic model, the authors use two variables to define the jobs' situation in each period (one variable ($y_{j,t}$) for identifying the time that the job begins to be processed and another one

($x_{j,t}$), for determining the periods that the job will be processed during them). Whereas, one of the assumption of this study is that the preemption of the jobs is not allowed. It means that when a job starts to be processed, the machine must continue the job until it be finished. So, we can use only one variable to define the jobs situation. Therefore, as a first contribution, we propose a more efficient model to formulate the problem with the least possible variables, easier to understand and easier to solve.

For this purpose, instead of using two variables ($x_{j,t}$ and $y_{j,t}$), the model is defined based on only one decision variable for the jobs ($y_{j,t}$).

$$y_{j,t} = \begin{cases} 1 & ; \text{ If job } j = 1, ..., n \text{ is performed in period } t \text{ by the machine} \\ 0 & ; \text{ Otherwise} \end{cases}$$

Since j can take n different values from 1 to n, and t can take $T + 1$ different values from 0 to T, this change will reduce $n * (T + 1)$ from number of variables. This value will be an impressive amount for large number of jobs and periods.

To formulate the new model, the equations where $x_{j,t}$ and $y_{j,t}$ appeared are changed. For this purpose, in (2.19) and (2.20), just $x_{j,t}$ is replaced by $y_{j,t}$, and constraints (2.21) and (2.22) replaced to (2.12) and (2.13), respectively. Moreover (2.23) supersedes to (2.14) and (2.15). The new equations are given in the following:

$$\sum_{j=1}^{n} y_{j,t} = \alpha_{1,t} \quad ;\forall\ t = 0, ..., T \tag{2.19}$$

$$\sum_{j=1}^{n} y_{j,t} \leq 1 \quad ;\forall\ t = 0, ..., T \tag{2.20}$$

$$p_j \cdot y_{j',t} \leq \sum_{t'=0}^{t} y_{j,t'} \qquad ;\forall\ t = 0, ..., T;\ j, j' = 1, ..., n;\ j' > j \tag{2.21}$$

$$\sum_{t'=1}^{t-p_j} y_{j,t'} + \sum_{t'=t+p_j}^{T} y_{j,t'} \leq T \cdot (1 - y_{j,t}) \qquad ;\ \forall\ t = 0, ..., T - p_j - 1;\ j = 1, ..., n \tag{2.22}$$

$$\sum_{t=1}^{T} y_{j,t} \geq p_j \quad ;\forall\ j = 1, ..., n \tag{2.23}$$

$$y_{j,t} \in \{0, 1\} \tag{2.24}$$

As in the basic model, (2.19) guarantees that the machine must be in state ON (s=1) when a job is processed. Equation (2.20) ensures that only one job can be processed during each period. Equation (2.21) allows a job to begin whereas, all of the precedent jobs in the given sequence have been completed. Equation (2.22) ensures the non-preemption of jobs. Finally, (2.23) forces all jobs to be processed completely during production shift.

2.3.3 Comparison

There are some criteria like objective value, the number of variables and constraints as well as computing time that can be considered to investigate the difference between these two mathematical models. Since, both of them are for the same problem, the objective value must be the same, and they have some difference in the other criteria.
To explore these differences, three different strategies and numerous examples based on each strategy are considered. All the instances are solved with both of the models using CPLEX software and on the same computer.

In the first strategy, processing time of each job is considered as an integer random number between 1 and 5 periods ($p_j \in [1, 5];\ \forall\ j = 1, ..., n$), besides, the cost of energy in each period is assumed as an integer random number between 1 to 10 ($c_t \in [1, 10];\ \forall\ t = 0, ..., T$). For examining the difference between these two model, twelve values are selected for the number of the jobs from 5 to 60, as well as the number of periods based on the other values was calculated and defined as input data for each example. After each test the objective values of both model in addition to computing time and difference between their number of variables are collected. Table 2.2 shows the obtaining results for the first strategy. Note that in this table B represents the basic model and I represents the improved model.
It must be mention that, the computation time for all of the instances is limited to one hour (3600 s).
As can be seen in Table 2.2, in the new model by eliminating one variable and rewriting some constraints, not only computing time and number of variables are significantly reduced (on average computation time 78.64% and number of variables 39.62%) but also, the amount of constraints

are decreased which can be very interesting specially for large number of the jobs. Therefore, it can be concluded that the performances of the model for solving the problem is increased. For example, within the limitation time, basic model is able to give optimal solution up to problem size of (30,107) whereas, our proposed model is able to present optimal solution up to problem size of (45,153).

In the second strategy, generally the assumptions are similar to the first one, just processing time of each job considered as an integer random number between 1 period and 10 periods ($p_j \in [1, 10];\ \forall\ j = 1, ..., n$). This change help us to explore the impact of having jobs with relatively long process time in the performance of these two models (Table 2.3). As we expected, this factor only increased the computation times, but actually it do not seem to have effect on the performance of these two models towards each other. As illustrated in Table 2.3, like the previous strategy, significantly decreases detected in computing time (83.76%) and number of variables (36.11%). For example, for the largest problem (problem with size of (25, 140)), for which the basic model is able to give the optimal solution in less than one hour, the basic model found the solution in 2785 seconds with 8743 variables, whereas, the improved one find the same result just in 29 seconds with 5218 variables.

In the third strategy, five examples have been generated by changing the energy costs of each period and the processing time of each job for each size of the problem. The results of this examination prove that the improvement in the computing times which is obtained by our model is not an accident. Among ten examples which basic model was not able to find the optimal solution in the limited time (3600s), the improved model found the optimal solution in five cases (Table 2.4).

(n, T)	Obj_B	Obj_I	CPU_B (s)	CPU_I (s)	$Gap_{Var}(\%)$
$(5, 23)$	322	322	0.18	0.12	22.68
$(10, 48)$	788	788	5.01	0.83	31.23
$(15, 59)$	1003	1003	13.21	1.78	35.70
$(20, 72)$	1213	1213	110.02	6.38	38.45
$(25, 92)$	1667	1667	1005.87	69.95	40.32
$(30, 107)$	1814	1814	2747.8	46.08	41.66
$(35, 126)$	-	2138	3600	491.34	42.68
$(40, 134)$	-	2504	3600	102.08	43.47
$(45, 153)$	-	2961	3600	2325.49	44.11
$(50, 172)$	-	-	3600	3600	44.64
$(55, 191)$	-	-	3600	3600	45.08
$(60, 213)$	-	4334	3600	3600	45.45

Table 2.2 – Comparative results for the first group of instances

(n, T)	Obj_B	Obj_I	CPU_B (s)	CPU_I (s)	$Gap_{Var}(\%)$
$(5, 35)$	580	580	0.33	0.21	22.70
$(10, 72)$	1197	1197	11.91	1.52	31.24
$(15, 101)$	1769	1769	94.44	14.37	35.71
$(20, 121)$	2285	2285	579.46	8.82	38.46
$(25, 140)$	2683	2683	2785.4	29.56	40.32
$(30, 170)$	-	3281	3600	117.18	41.66
$(35, 209)$	-	4452	3600	3600	42.68

Table 2.3 – Comparative results for the second group of instances

(n, T)	Obj_B	Obj_I	Gap_{Obj}	Gap_{CPU} (%)	$Cons_B$	$Cons_I$	Gap_{Cons}
$(25, 110)$	1645	1522	123	82.31	40417	40268	149
$(25, 110)$	1738	1580	158	51.52	40417	40272	145
$(25, 110)$	1544	1533	11	17.58	40417	40273	144
$(25, 110)$	1560	1315	245	88.03	40417	40272	145
$(25, 110)$	1421	1275	146	0.00	40417	40279	138
$(30, 130)$	2086	1930	156	0.00	66047	65871	176
$(30, 130)$	2412	2195	217	27.58	66047	65861	186
$(30, 130)$	2125	1968	157	0.00	66047	65867	180
$(30, 130)$	2115	1665	450	0.00	66047	65880	167
$(30, 130)$	2265	2021	244	0.00	66047	65865	182

Table 2.4 – Comparative results for the third group of instances

2.4 Complexity analysis

The experimental results in [61], show the disability of the presented mathematical model to solve the medium and large size instances of this problem (more than 60 jobs) in a reasonable time. In their work, the authors mentioned that "because the shop floor scheduling problem is considered to be an NP-hard-complete problem, the formulated problem in their paper cannot be solved in real life". So, they used a meta heuristic approach (genetic algorithm) to obtain a feasible solution in a reasonable computational time.
In this section, we prove that it is possible to solve optimally this problem with a polynomial algorithm which is based on the dynamic programming approach. For this purpose, first of all, the steps for the graph construction are described and then we discussed the complexity of the research of the shortest path in the graph.

2.4.1 Dynamic programming approach

In this section, the addressed problem is modeled with a graph and then, a dynamic programming approach is used to find the optimal solution in a reasonable time. For this purpose, a finite graph whose dimensions (number of vertices and edges) dependent on the total processing times and the total number of periods is used to model the problem.
In the following, the presented approach is described with more details.

Graph construction steps

In the considered problem, which is characterized by a fixed number of periods (T), a minimum number of periods are required to accomplish the necessary activities. These activities include turning on and turning off the machine for at least one time, and performing all the jobs. The number of required periods for performing all the jobs (P) is equal to sum of the processing times ($P = \sum_{j=1}^{n} p_j$). Moreover, at least $\beta_1 + \beta_2 + 1$ periods are required for initial turning on, final turning off and final OFF states (if only once switch on and off to be considered). During the remaining periods (the difference between the total number of periods and the minimum required number of periods), the machine must be in non-processing states, i.e. initial OFF, middle OFF, final OFF, and Idle states. Note that each middle-OFF state consists of a sequence of Toff, OFF

during at least one period, and Ton states.

Let x indicates the number of extra periods, so:

$$x = T - P - (\beta_1 + \beta_2 + 1) \tag{2.25}$$

For example, we consider an instance with $P = 5, T = 15, \beta_1 = 2, \beta_2 = 1$, where x value is equal to [15-(2+1+1)-5]=6 periods.

Based on the problem's objective, these x periods can be allocated to a combination of initial or final OFF states, Idle states between the ON states, and middle OFF states.

In this section, a graph consisting several decision-making levels is considered to model the problem. Each level represents one period of the horizon time. So, the graph consists of $T + 1$ decision levels ($0 \leq l \leq T$). This approach can be applied in three steps as follows.

Step 1: putting the nodes For each decision level, let us consider H_l consisting the possible nodes for level l, that corresponds to the different machine states. Therefore, each node of H_l is characterized by the cumulative number of production units (k) from period 0 to l. Since, the initial and final states of the machine are considered as OFF state, $H_0 = \{I\}$ and $H_T = \{F\}$, where I represents the initial state of the machine, and F represents the machine's final status after processing all the jobs (P).

As an example, for the presented problem in Figure 2.7, the possible number of production units until period 4 can be 1 or 2 units ($H_4 = \{1, 2\}$). Moreover, because of the value of x for this instance, the machine can be in the initial state during period 4. Also, the possible number of production units until period 7 can be from 1 to 5 units ($H_7 = \{1, 2, 3, 4, 5\}$). If it was less than 1 unit, there will not be enough time to complete all the jobs in the future. Moreover, based on the set up times, 7 periods are not enough to process all the 5 production units and turn off the machine.

The earliest and the latest possible periods (or levels in our graph's definition) for any node k depends on the x value of the given problem. These levels can be shown by the interval of $\tau_k = \{l_{min(k)}, \cdots, l_{max(k)}\}$ in the scheduling horizon.

For example, the latest period for the machine to be in the initial OFF state (I), is when enough

periods remain to perform the necessary setups and complete the processing jobs (x). The time interval for node k is:

$$\{l_{min(k-1)} + \beta_{k-1,k} + 1, \cdots, l_{max(k-1)} + \beta_{k-1,k} + 1\} \tag{2.26}$$

where $\beta_{k-1,k}$ is the required number of periods for the transition from $k-1$ to k. As it is mentioned before, in this problem, just the transitions between the OFF state and the first job, besides the transition between the last job and OFF state are considered. So, $\beta_{k-1,k} = 0;\ \forall\ k \in \{2, \cdots, P\}$, and the time interval for node k is simplified as:

$$\tau_k \in \{\beta_1 + k, \cdots, x + \beta_1 + k\} \quad ; \forall k \in \{1, \cdots, P\} \tag{2.27}$$

The first step of this dynamic programming approach is to put all the nodes ($k \in \{I, 1, \cdots, P, F\}$) in the graph using their related time interval (τ_k).
Therefore, the total number of nodes (vertices) (V) for the presented graph is:

$$|V| = P \times (x + 1) + 2 \cong TP \tag{2.28}$$

To illustrate different graph construction steps, the corresponding graph for the considered example is presented at the end of each step. So, the first step of the related graph for this example is presented in Figure 2.3 .

Step 2: drawing the edges The second step is to draw the edges of the graph and compute their values. In this approach, the edges of the graph represent the possible transitions between two nodes (node (k, l) and node (k', l')). Moreover, the edges are valued by the amount of total energy consumption costs for performing the related transition, which are the positive values ($Ev_{(k,l)-(k',l')};\ \forall\ k \in H_l,\ k' \in H_{l'},\ l' \geq l + 1$).
Because of the different possible transitions between the nodes, different types of the edges exist. They can be divided into three main sets ($E_1,\ E_2,\ E_3$).

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

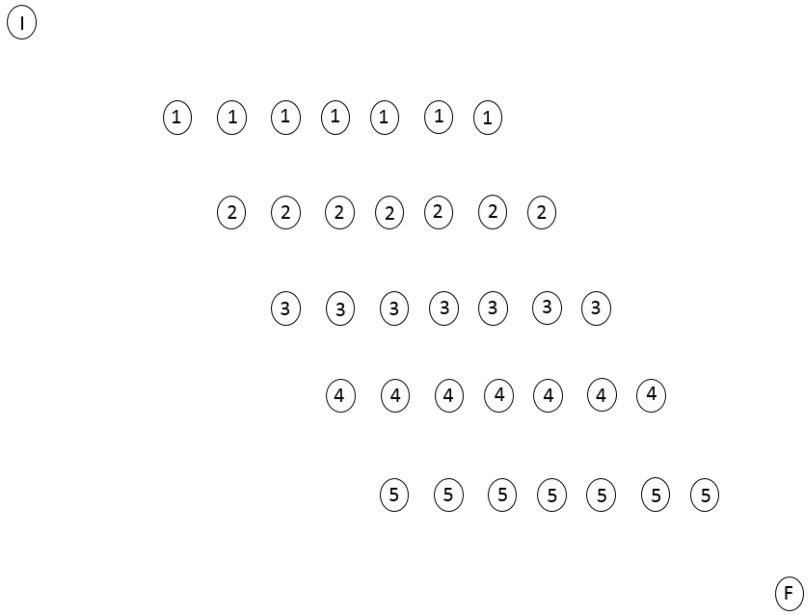

Figure 2.3 – Graph construction: step 1

Step 2.1: E_1 The first set (E_1) connects the nodes with the same cumulative production number k between level l and $l+1$, which indicates the Idle states. The edge's value of this set is:

$$Ev_{(k,l)-(k,l+1)} = c_{l+1} \times e_{Idle} \quad ;\forall k \in \{p_1, p_1+p_2, \cdots, \sum_{j=1}^{n-1} p_j\} \tag{2.29}$$

Where c_l is the unit of energy cost in period l, e_{Idle} is the machine's energy consumption in $Idle$ state, and p_j is the process time of job j. The total number of these edges is equal to $|E_1| = (n-1) \times x$.

The first part of the second step of the related graph for the considered example is presented in Figure 2.4 .

Step 2.2: E_2 The second set of edges (E_2) connects nodes k in level l to node $k+1$ in level l', which illustrates three transition cases. The first case is for initial turning on phase, with the

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

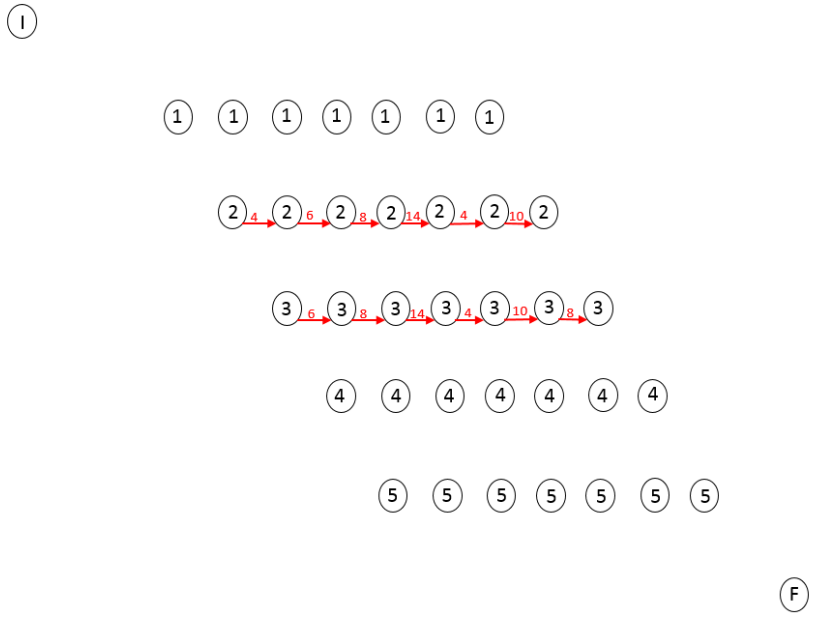

Figure 2.4 – Graph construction: step 2.1

edge value of:

$$Ev_{(I,0)-(1,l)} = \sum_{i=l-\beta_1}^{l-1} (c_i \times e_{Ton}) + c_l \times e_{ON} \tag{2.30}$$

The second case is for processing the next job with the edge value of:

$$Ev_{(k,l)-(k+1,l')} = c_{l'} \times e_{ON} \quad ; l' = l + 1 \tag{2.31}$$

And the third case is for final turning off with the edge value of:

$$Ev_{(P,l)-(F,T)} = \sum_{i=l+1}^{l+\beta_2} (c_i \times e_{Toff}) + \sum_{i=l+\beta_2+1}^{T} (c_i \times e_{OFF}) \tag{2.32}$$

The cardinal of these edges is equal to $|E_2| = (P + 1) \times (x + 1)$.

The second part of the second step of the related graph for the considered example is presented in Figure 2.5.

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

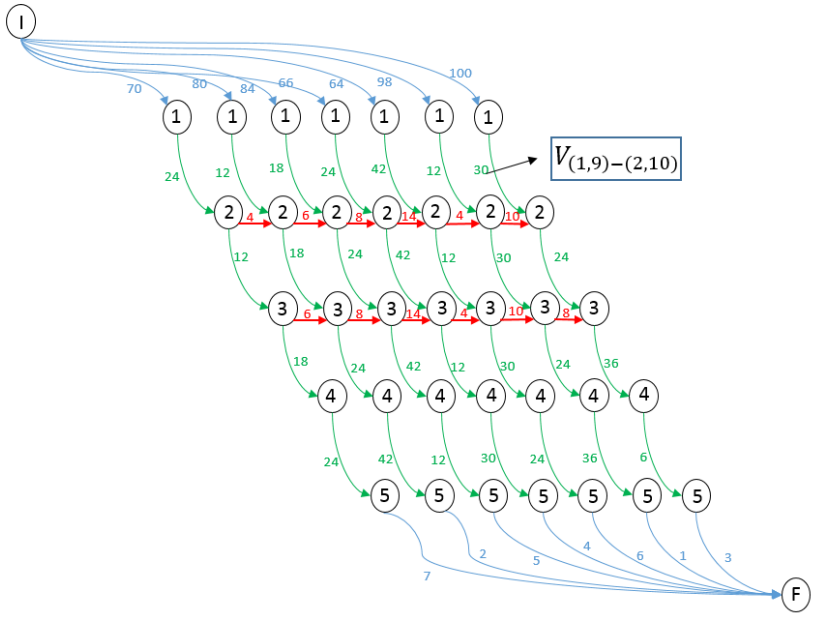

Figure 2.5 – Graph construction: step 2.2

Step 2.3: E_3 The last set of edges (E_3) shows the middle-OFF states between two processing jobs, which connects nodes k in level l' with node $k+1$ in level l, where, $l' \in \{l_{min(k)}, \cdots, x + k - \beta_2 - 1\}$. The edge's value of this set is:

$$Ev_{(k,l')-(k+1,l)} = \sum_{i=l'+1}^{l'+\beta_2}(c_i \times e_{Toff}) + \sum_{i=l-\beta_1}^{l-1}(c_i \times e_{Ton}) + c_l \times e_{ON} \quad ; \quad \forall k \in \{p_1, p_1+p_2, \cdots, \sum_{j=1}^{n-1} p_j\} \tag{2.33}$$

The cardinal of these edges is equal to:

$$|E_3| = \sum_{i=1}^{x-(\beta_1+\beta_2)} i \times (n-1) = \frac{(x-(\beta_1+\beta_2)) \times (x-(\beta_1+\beta_2)+1)}{2} \times (n-1) \tag{2.34}$$

Therefore, the total number of edges (E) for the presented graph is:

$$|E| = |E_1| + |E_2| + |E_3| \cong T^2 P \tag{2.35}$$

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

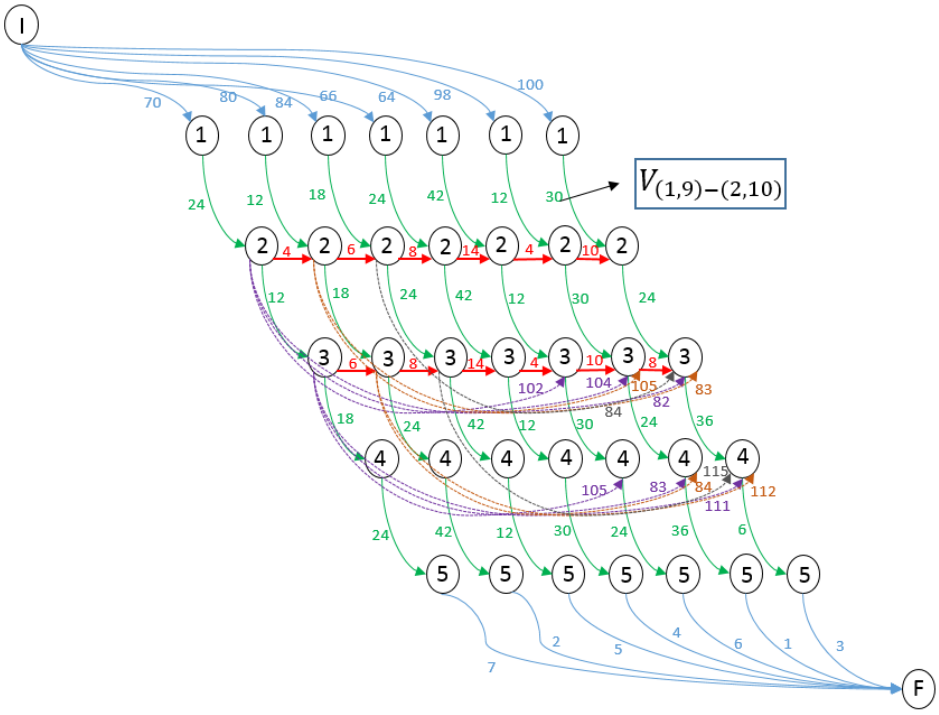

Figure 2.6 – Graph construction: step 2

The corresponding graph at the end of the second step for the considered example consists of 37 vertices and 66 edges which is presented in Figure 2.6 .

2.4.2 The shortest path

The third step of this approach is to find the optimal solution of the problem. Based on the graph modeling approach, each path from node I of level 0, to node F of level T represents a feasible solution of the problem and its weight represents the total energy consumption costs. Since the objective is to minimize the total energy costs, the shortest path that starts at node $(I, 0)$ and ends in node (F, T) represents the optimal solution of the problem.

Recurrence formulation

Until here, in our approach, we present a graph modeling which is able to model all the possible solutions of the problem in a polynomial time. In this section, our aim is to prove that the optimal solution of this problem can be obtained in a polynomial time too. All the graph's edge values are positive in this approach. So, Dijkstra's algorithm, which is one of the most efficient algorithms to find the shortest path between the source node and every other node of a graph, is applicable for the presented approach.

For this purpose, let us consider the cost $C_{(k,l)}$ associated to node $k \in H_l$ which indicates the minimum cost to set production level k at period l. The recurrence relationship for evaluating the cost of each node, is as follows:

$$\begin{aligned} &C_{(I,0)} = 0 \\ &C_{(k,l)} = \min_{(k',l') \in A_{k,l}} \{C_{(k',l')} + Ev_{(k',l')-(k,l)}\}; \quad \forall k \in \{1, \cdots, P, F\}; \quad \forall l \in \{1, \cdots, T\} \end{aligned} \tag{2.36}$$

where $A_{k,l}$ is set of the precedent nodes that are connected to node k of period l directly. For example, in Figure 5.4, $A_{4,11} = \{(3,10),(3,6),(3,5)\}$ and $C_{(4,11)}$ obtains from the following calculations:

$$\begin{cases} C_{(3,10)} + Ev_{(3,10)-(4,11)} = 126 + 24 = 150 \\ C_{(3,6)} + Ev_{(3,6)-(4,11)} = 110 + 84 = 194 \\ C_{(3,5)} + Ev_{(3,5)-(4,11)} = 106 + 83 = 189 \\ C_{(4,11)} = \min\{150, 194, 189\} = 150 \end{cases} \tag{2.37}$$

Finally, $C_{(F,T)}$ represents the optimal value of the objective function for the considered problem. The application of Dijkstra's algorithm to the assumed problem when its parameters' values are considered as in Table 2.5, is illustrated in Figure 2.7. The shortest path is $(0-1-2-3-4-5-F)$ with the cost of 155. In other words, the best solution for this example is to turn on the machine from period 0, then to process all the jobs based on their order, during periods 3 to 7, and finally,

Machine's state	Power consumption (kW)	Required time (period)
ON	6	5={2,1,2}
OFF	0	-
Idle	2	-
Ton	8	2
Toff	1	1

Table 2.5 – Parameter values of example (5,15)

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

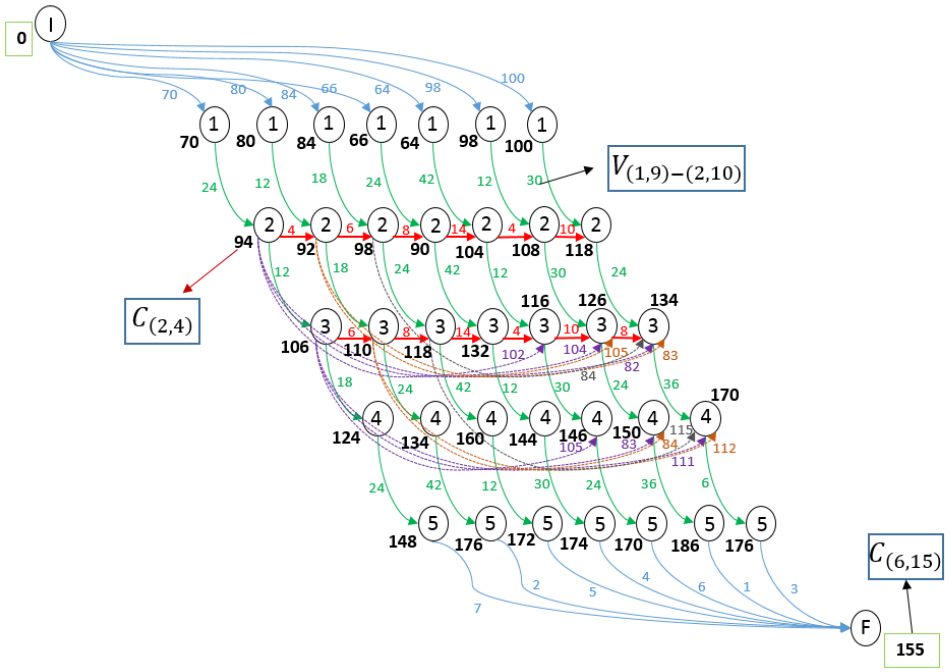

Figure 2.7 – Graph representation for the problem with fixed sequence

to turn the machine off in period 9.

After describing the new formulation approach, the complexity of Dijkstra's algorithm for this problem is evaluated in the following section.

2.4.3 Dynamic programming complexity

According to [77], the worst case implementation of Dijkstra's algorithm is based on a min-priority queue, that runs in $O(|E| + |V| \log |V|)$ (where $|E|$ is the number of edges and $|V|$ is the number of vertices or nodes). Consequently, the complexity of this algorithm for the presented Dynamic

programming approach is equal to:

$$O(T^2P + TP\log TP) = O(T^2P + TP\log T + TP\log P) \cong O(T^2P) \tag{2.38}$$

Since the largest possible value of P is T (worst case analysis), the considered problem is polynomial of degree 3 or a cubic polynomial problem ($O(T^3)$). Note that, in this study, Dijkstra's algorithm is chosen, while in the literature there exist some other algorithms which can find the shortest path more faster. But, since our objective is to analyze the complexity of this problem, we did not verify the other algorithms. Whereas, it is possible to optimize the computation time of this approach by using another algorithm.

In the following, some numerical experiments are presented to investigate the effectiveness of the proposed dynamic modeling approach for the considered problem.

2.4.4 Numerical experiments

In this section, the proposed approach in this study and the presented mathematical model in [61], are implemented and tested on several instances. The linear programming method is implemented on ILOG CPLEX Software, and dynamic programming approach is coded with the C++ programming language in Visual Studio 2015. A computer with 2.6 GHz Intel Core i5 processor and 8 GB of RAM is used to implement all the experiments. At first, the 4 cases presented in [61], are implemented by these two methods (see Table 2.6). The energy prices during the periods and the optimal solutions of these case are given in figures 2.8 to 2.11. For all these examples, our approach finds the optimal solution in less than one second. Since, in [61], the comparison between the analytical solution and the genetic algorithm is evaluated for a problem as large as $(60, 135)$, in this paper, several examples with sizes from $(5, 23)$ to $(60, 213)$ are examined. For this purpose, the number of periods, number of jobs and their processing times, as well as energy costs in each period, are generated randomly for every instance. Computation times are limited to 2 hours ($7200s$) for the proposed model by [61]. Table 2.7 represents a part of the results obtained for the problem with P process times and T periods, that are compared in terms of costs obtained and computation times. As the results show, since this problem is polynomial, in all the cases our approach finds the optimal solution in a few seconds. Whereas,

Period numbers	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Price per period	8	8	8	4	4	4	3	3	3	2	2	2	2	2	10	10	10	10	3	3	3	2	2	2	2	6	6	3	3	3	5	5
Processing Jobs						16	12	12	12	8	8	8	8	8							12	8	8	8	8							
Idle (I)																																
Turning ON (T_{on})				20	20														15	15												
Turning OFF(T)															10											6						
Shutdown (S)	0	0	0													0	0	0									0	0	0	0	0	0
The Schedule	S			T_{on}		Job1			Job2			Job3			T	S			T_{on}		Job4			Job5		T	S					

Figure 2.8 – The energy prices during the periods and the optimal solution for the first case ([61])

Period numbers	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Price per period	2	2	2	2	3	3	3	3	3	8	8	2	2	2	2	2	5	5	5	5	5	5	5	2	2	3	3	3	2	2	2	2	5	5	5	5	3	3	2	2	2	2	2	3	3	5	5
Processing Jobs			8	8	12	12	12							8	8	8										12	12	12	8	8	8	8	20	20	20	20	12	12	8	8	8	8	8	8	12	12		
Idle (I)																																																
Turning ON (T_{on})	10	10										10	10												10	10																						
Turning OFF(T)								3									5																														5	
Shutdown (S)									0	0	0							0	0	0	0	0	0																									0
The Schedule	T_{on}		Job1		Job2		T		S			T_{on}		Job3	T		S							T_{on}		Job4			Job5			Job6			Job7			Job8			Job9	Job10	T	S

Figure 2.9 – The energy prices during the periods and the optimal solution for the second case ([61])

Period numbers	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Price per period	2	2	2	2	2	3	3	3	3	7	7	2	2	2	2	2	3	3	3	3	6	6	6	6	2	2	2	2	2	2	3	3	3	3	7	7	7	7	7	5	5	5	5	5	6	6	6	6
Processing Jobs			8	8	8	12	12	12	12	28		8	8	8	8	8	12	12	12	12	24	24		24	8	8	8	8	8	8	12	12	12															
Idle (I)											14												12																											
Turning ON (T_{on})	10	10																																															
Turning OFF(T)																																		3															
Shutdown (S)																																			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
The Schedule	T_{on}		Job1		Job2		Job3		I		Job4			Job5			Job6		I		Job7		Job8		Job9	Job10	T	S																	

Figure 2.10 – The energy prices during the periods and the optimal solution for the 3rd case ([61])

Period numbers	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Price per period	2	2	2	2	2	3	3	3	3	3	3	2	2	2	2	2	3	3	3	3	3	6	6	6	6	3	3	3	3	3	2	2	2	2	2	70	70	70	10	8	8	8	8	6	6	6	6
Processing Jobs			8	8	8	12	12	12	12	12	12	8	8	8	8	8	12	12	12	12	12					12	12	12	12	12	8	8	8	8	8													
Idle (I)																						12	12	12	12																								
Turning ON (T_{on})	10	10																																															
Turning OFF(T)																																				2													
Shutdown (S)																																					0	0	0	0	0	0	0	0	0	0	0	0	0
The Schedule	T_{on}		Job1		Job2		Job3		Job4			Job5			Job6			I				Job7		Job8		Job9	Job10	T	S																	

Figure 2.11 – The energy prices during the periods and the optimal solution for the 4th case ([61])

for the instances larger than $(30, 170)$, CPLEX solver is not able to find any solution for these problems using the presented model in [61], within the 2 hours time limitation.

	(P,T)	Obj_{Cplex}	$CPU_{Cplex}(s)$	Obj_{DP}	$CPU_{DP}(s)$
Case1	(5,32)	222	0.61	222	0
Case2	(10,48)	393	3.57	393	0
Case3	(10,48)	393	1.15	393	0
Case4	(10,48)	366	0.97	366	0

Table 2.6 – Comparison between Cplex and DP approach solutions for the presented cases in [61]

(P,T)	Obj_1	$CPU_1(s)$	Obj_2	$CPU_2(s)$
(5,23)	322	0.12	322	0
(5,35)	580	0.91	580	0
(10,48)	789	1.73	789	0
(10,72)	1239	4.2	1239	0.01
(15,59)	1032	4.88	1032	0
(15,101)	1818	24.05	1818	0.01
(20,72)	1248	14.30	1248	0
(20,121)	2327	22.58	2327	0.01
(25,92)	1667	29.54	1667	0.03
(25,140)	2725	40.51	2725	0.01
(30,107)	1871	241.05	1871	0.03
(30,170)	3337	390.81	3337	0.11
(35,126)	2204	1557.47	2204	0.09
(35,209)	-	7200	4197	0.14
(40,134)	2565	293.08	2565	0.03
(45,153)	2989	5524.68	2989	0.08
(50,172)	-	7200	3443	0.06
(55,191)	-	7200	3853	0.09
(60,213)	-	7200	4291	0.19

Table 2.7 – Comparison between Shrouf et al.'s model (1) and our approach (2)

2.5 Conclusion

In this chapter, the scheduling problem of several jobs with a fixed sequence on a multi-states single machine $(1, TOU|states,\ sequence|TEC)$ is addressed. As a first contribution, an improved LP model is proposed based on the Shrouf et al.'s one. The presented model is formulated by using one decision variable for the jobs, so, the number of variables are reduced comparing to the basic model. Moreover, by rewriting some constraints, computing times and number of constraints are decreased. Therefore, the performance of the mathematical model to solve the problem is increased. For example, within the same limitation time, basic model is able to give optimal solution up to problem size of (30,107) whereas, the improved model is able to present the optimal solution up to problem size of (45,153).

A new dynamic programming approach is proposed to model the problem by using a finite graph. Then, the Dijkstra's algorithm is used to find the shortest path of the graph. After analysing the complexity of this approach, it is proved that unlike what is considered in [61] the problem is polynomial.

After analysing the complexity of this problem, it is very interesting to analyse the complexity of the general version of this problem without fixed sequence of the jobs $(1, TOU|\ states|TEC)$. For this purpose, in the next chapter, the complexity of this problem and some other sub problems, is studied.

3

Multi-states single-machine energy-efficient scheduling problem: general version

Outline of the current chapter

3.1 Introduction

Shrouf et al.'s study [61] and previous chapter of this thesis deal with the problem in which the jobs must be done in a fixed sequence and their order is not changeable. So, the presented models just specify the optimal machine's state in each period based on the total electricity cost minimization. Whereas, in most manufacturing industries, finding the optimal sequence of processing jobs is an important issue for manufacturers that can cause energy cost minimization. To the best of our knowledge, there is no study that considers the energy efficient scheduling problem of a single machine system as defined here for finding optimal job's sequence. Therefore, in this chapter, a generalization of the previous problem is proposed to find the optimal sequences of the machine's states and the jobs simultaneously. A new mathematical model is presented for this problem, its complexity is analyzed and some resolution methods are proposed. Moreover, the complexity of several sub problems are analyzed.

The remainder of this chapter is organized into seven sections. In section 3.2, the definition of the problem with its assumptions and constraints are presented. In section 3.3, a new model for the considered problem is presented. Moreover, the comparison between this problem and the studied problem in chapter 2 is also represented. In section 3.4, the complexity of this problem is analyzed by using 3-PARTITION problem. In section II, several lower bounds are proposed for the problem. In section II, a heuristic algorithm and a genetic algorithm are presented to solve the problem. In section II, the complexity of several sub problems are studied. Finally, the brief conclusion of this chapter is presented in section 3.8.

Parameter	Value	Parameter	Value
β_1	2	β_2	1
p_1	3	p_2	2
p_3	4	p_4	2
p_5	3	e_{OFF}	0
e_{ON}	4	e_{Idle}	2
e_{Ton}	5	e_{Toff}	1

Table 3.1 – The parameters' values for an instance with 5 jobs and 32 periods

3.2 Problem presentation

The problem addressed in this chapter can be described as follows. It deals with the scheduling problem of several jobs on a multi-states single machine without fixed sequence $(1, TOU|states|TEC)$. So, we keep all the assumptions and constraints of chapter 2, and we just relax the constraint that the jobs must be processed in a given order. By this way, the machine can process the jobs in a economic way and change the machine's state regarding to their energy consumption costs.

3.2.1 Illustrative example

Let us consider an example with 5 jobs, 32 periods and the parameters' values as Table 3.1. The optimal solution of this instance for a problem with a fixed sequence for the jobs is presented in Figure 3.1, and for a problem without a fixed sequence is presented in Figure 3.2. As it can be seen, when the jobs must be processed in a predetermined order (Figure 3.1), the optimal solution has the total energy cost of 235. It can be obtained by turning on the machine at period 5 and 6, process the job1, job2 and job3 during periods 7 to 15, turn it off for periods 16 to 20, process job4 and job5 just after, and finally, turn the machine off from period 26. When the processing order of the jobs is flexible, the optimal solution has different machine's state and jobs' sequence during the horizon which is less expensive. The total energy cost of 234 can be obtained by turning the machine on at period 8 and 9, process job2, job4 and job1 during the periods 10 to 16, Idle state in periods 17 and 18, process the job5 and job3 just after, and finally, turn the machine off at period 26.

Period number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	Cost
Energy price	0	8	5	5	8	4	4	4	3	3	3	2	2	4	2	2	7	7	10	3	3	3	2	2	4	2	6	6	3	3	3	5	5	
ON (s=1)								16	12	12	12	8	8	16	8	8						12	8	8	16	8								
OFF (s=2)	0	0	0	0	0													0	0									0	0	0	0	0	0	
Idle (s=3)																																		
Turn on (s=4)						20	20													15	15													
Turn off (*s=5*)																	7										6							
The schedule	s=2					s=4		Job1			Job2		Job3				s=5	s=2		s=4		Job4		Job5			s=5	s=2						235

Figure 3.1 – Optimal solution of the example with fixed sequence

Period number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	Cost
Energy price	0	8	5	5	8	4	4	4	3	3	3	2	2	4	2	2	7	7	10	3	3	3	2	2	4	2	6	6	3	3	3	5	5	
ON (s=1)											12	8	8	16	8	8	28			12	12	12	8	8	16	8								
OFF (s=2)	0	0	0	0	0	0	0	0																				0	0	0	0	0	0	
Idle (s=3)																		14	20															
Turn on (s=4)									15	15																								
Turn off (*s=5*)																											6							
The schedule	s=2								s=4		Job2		Job4			Job1		s=3		Job5			Job3			s=5	s=2						234

Figure 3.2 – optimal solution of the example without fixed sequence

3.3 Mathematical model

The mathematical model of this problem can be obtained from the previous model, by relaxing the constraint (equation 2.21) that forced the machine to process the first job in the given sequence completely and then process the next jobs.

$$Pb_2 = Min \sum_{t=0}^{T} c_t \left(\sum_{s=1}^{k} e_s \cdot \alpha_{s,t} + \sum_{s=1}^{k} \sum_{s'=1}^{k} e_{ss'} \cdot \beta_{ss',t} \right) \tag{3.1}$$

$$\sum_{j=1}^{n} y_{j,t} = \alpha_{1,t} \quad ; \forall t = 0, ..., T \tag{3.2}$$

$$\sum_{s=1}^{k} \alpha_{s,t} + \sum_{s=1}^{k} \sum_{s'=1}^{k} \beta_{ss',t} = 1 \qquad ; \forall t = 0, ..., T \tag{3.3}$$

$$\alpha_{s,t} \leq \sum_{s'=1/d_{ss'}=0}^{k} \alpha_{s',t+1} + \sum_{s=1/d_{ss''}\geq 1}^{k} \sum_{s''=1}^{k} \beta_{ss'',t+1} \qquad ; \forall t = 0, ..., T-1; s = 1, ..., k \tag{3.4}$$

$$\beta_{ss',t} \leq \beta_{ss',t+1} + \alpha_{s',t+1} \qquad ; \forall t = 0, ..., T-1; s, s' = 1, ..., k; d_{ss'} \geq 1 \tag{3.5}$$

$$\sum_{p'=p+1}^{t+d_{ss'}} \beta_{ss',t'} \geq (\alpha_{s,t} + \beta_{ss',t+1} - 1) \cdot d_{ss'} \qquad ; \forall t = 0, ..., T-1; s, s' = 1, ..., k; d_{ss'} \geq 1 \tag{3.6}$$

$$\beta_{ss',t} + \beta_{ss',t+d_{ss'}} \leq 1 \qquad ; \forall t = 0, ..., T - d_{ss'}; s, s' = 1, ..., k; d_{ss'} \geq 1 \tag{3.7}$$

$$\sum_{j=1}^{n} y_{j,t} \leq 1 \quad ;\forall t = 0, ..., T \tag{3.8}$$

$$\sum_{t'=1}^{t-p_j} y_{j,t'} + \sum_{t'=t+p_j}^{T} y_{j,t'} \leq T \cdot (1 - y_{j,t}) \quad ;\forall t = 0, ..., T - p_j - 1; j = 1, ..., n \tag{3.9}$$

$$\sum_{t=1}^{T} y_{j,t} \geq p_j \quad ;\forall j = 1, ..., n \tag{3.10}$$

$$\alpha_{2,t} = 1 \qquad ;t = \{0, T\} \tag{3.11}$$

$$\alpha_{s,t},\ \beta_{ss',t} \in \{0, 1\} \tag{3.12}$$

$$y_{j,t} \in \{0, 1\} \tag{3.13}$$

In this model, 3.1 deals with the objective function which computes the energy consumption costs depending on the machine status and the energy prices at each period. Equation (3.2) ensures that the machine is able to process a job, just when the machine is in processing state. Equation (3.3) shows that at each period the machine must be in one of the states or in a transition between two states. Equations (3.4) and (3.5) limit the state of the machine at one period based on the status or transition that the machine has in the previous period. Equations (3.6) and (3.7) identify lower and upper number of periods in which the machine can be in a transition state to perform the jobs before the time limitation. Equation (3.8) specifies that the machine can process only one job during each period. Equation (3.9) imposes the non-preemption of jobs. Equation (3.10) introduces the processing time of each job and forces all jobs to be completed during the time limitations. Equation (3.11) presents the boundary conditions.

3.4 Complexity analysis

One of the first step for investigating a new problem is to analyze its complexity to know that the problem is an NP-hard problem or a polynomial one. In the following, it is attempted to prove the NP-hardness of this problem by using a 3-PARTITION problem.

Theorem 1 ***If the sequence of the jobs not be fixe, the Problem*** $(1, TOU|states|TEC)$ ***is strongly NP-hard.***

Proof. The proof is based on the fact that the decision problem related to this optimization problem, may be reduced to a 3-PARTITION problem, which is strongly NP-hard ([14]).
Given positive integers $\{a_1, a_2, \cdots, a_{3t}, b\}$, such that:

$$b/4 < a_j < b/2 \quad ;\forall j = 1, 2, \cdots, 3t \tag{3.14}$$

$$\sum_{j=1}^{3t} a_j = tb \tag{3.15}$$

The following instance (equations (3.16) to (3.19)), with $n = 3t$ jobs and $T = tb + t + 3$ periods can be constructed. The machine consumes the units of energy just when it is in ON state ($e_{ON} \neq 0;\ e_{OFF} = e_{Idle} = e_{Ton} = e_{Toff} = 0$). Moreover, the unit of energy price in some periods (tb periods) equals to 0, and for the rest ($t + 3$ periods) is equal to c ($c > 0$):

$$p_j = a_j \quad ;\forall j = 1, 2, \cdots, 3t \tag{3.16}$$

$$c_t = c \quad ;\forall t = 0, 1, 2, tb + t + 3, (i + 1)b + i + 3 \quad ;\forall i = 0, 1, \cdots, t - 1 \tag{3.17}$$

$$c_t = 0 \quad ;\forall t = i(b + 1) + 3, \cdots, i(b + 1) + (b - 1) + 3 \quad ;\forall i = 0, 1, \cdots, t - 1 \tag{3.18}$$

$$e_{OFF} = e_{Ton} = e_{Idle} = e_{Toff} = 0, \quad e_{ON} = \varepsilon \tag{3.19}$$

Let us consider a decision problem that searches a solution such that it's total energy consumption costs is equal to 0. A schedule with total energy costs of 0 ($TEC = 0$), exists if and only if, the machine is in one of the states that consume 0 unit of energy during the periods with $c_t = c$, and it is in state ON when $c_t = 0$. This can be achieved if and only if, all the $3t$ jobs are partitioned over the t intervals with the length of b periods. For this purpose, the $3t$ jobs must be partitioned to t sets such that each set consists of 3 jobs, and the sum of their processing times must be equal to b. Then, each set must be partitioned into one interval with the length of b periods, which can be achieved if and only if, 3-PARTITION has a solution (see Fig. 3.3). Therefore, since the 3-PARTITION is known as an NP-complete problem ([14]), so, $1, TOU|states|TEC$ is an NP-hard problem. ■

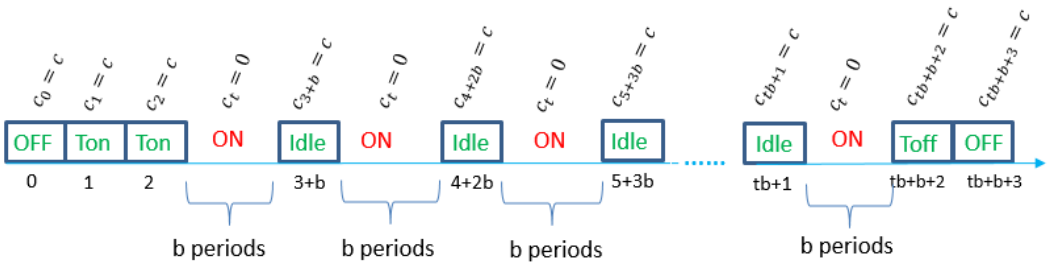

Figure 3.3 – An example of $1, TOU|states|TEC$ problem which transfers to a 3-PARTITION problem

3.5 Complexity analysis of some variants of the problem

In this section we are attempted to investigate the complexity of some variants of the problem. In this study, we tried to keep the main assumptions as the general problem for all the subproblems and change just one of them in each problem.

Let consider φ_s as the set of periods' number in which the machine is in state $s \in \{OFF,\ Ton,\ ON,\ Toff,\ Idle\}$, and Z which represents the objective value of any feasible solution. Besides, Z^* and φ_s^* are related to the optimal solution. So, the objective value of each subproblem may be computed with the following formulation:

$$\begin{aligned} Z = &(e_{OFF} * \textstyle\sum_{t\in\varphi_{OFF}} c_t) + (e_{Ton} * \sum_{t\in\varphi_{Ton}} c_t) + \\ &(e_{ON} * \textstyle\sum_{t\in\varphi_{ON}} c_t) + (e_{Toff} * \sum_{t\in\varphi_{Toff}} c_t) + (e_{Idle} * \sum_{t\in\varphi_{Idle}} c_t) \end{aligned} \tag{3.20}$$

Since the initial and final states of the machine are assumed as OFF states for all the problems, the machine is in Ton/Toff state at least for once. Let consider λ which is a positive integer number ($\lambda \geq 1$) to indicate the number of Ton or Toff states over the T periods, so:

$$|\varphi_{Ton}| = \lambda * \beta_1 \tag{3.21}$$

$$|\varphi_{Toff}| = \lambda * \beta_2 \tag{3.22}$$

It must be mention that for each additional Toff/Ton transitions, the machine must stay in OFF state during at least one period. It means that ($|\varphi_{OFF}| \geq \lambda$).

As it is described in chapter 2, in any feasible solutions for these kind of problems, there are always

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	Cost
c_t	0	8	8	8	4	4	4	3	3	3	2	2	2	2	2	10	10	10	10	3	3	3	2	2	2	2	6	6	3	3	3	5	5	
ON							16	12	12	12	8	8	8	8	8							12	8	8	8	8									
OFF	0	0	0	0													0	0	0									0	0	0	0	0	0	
Idle																																		
Turn on					20	20														15	15													
Turn off																10											6							
The schedule	Off				Ton		Job1			Job2		Job3				Toff	Off			Ton		Job4		Job5			Toff	Off						222

Figure 3.4 – An example for the general problem

some extra periods (denoted by x in equation 5.1) that the machine must be in non-processing states during them.

Therefore, the cardinal of sets φ_s ; $\forall s \in \{Ton,\ ON,\ Toff,\ Idle,\ OFF\}$ in any feasible solution are as follow:

$$\begin{cases} |\varphi_{Ton}| = \lambda * \beta_1 \\ |\varphi_{ON}| = P \\ |\varphi_{Toff}| = \lambda * \beta_2 \\ |\varphi_{Idle}| \geq 0 \\ |\varphi_{OFF}| \geq \lambda \end{cases} \tag{3.23}$$

where:

$$|\varphi_{Ton}| + |\varphi_{ON}| + |\varphi_{Toff}| + |\varphi_{Idle}| + |\varphi_{OFF}| = T \tag{3.24}$$

Since in each feasible solution, all the jobs must be processed completely, we have $|\varphi_{ON}| = P$, so:

$$|\varphi_{Ton}| + |\varphi_{Toff}| + |\varphi_{Idle}| + |\varphi_{OFF}| = T - P \tag{3.25}$$

The gant chart for an instance of 5 jobs and 32 periods with the parameters' values as Table 3.1 is provided in Fig. 3.4.

For this instance, the value of λ is equal to 2 ($|\varphi_{OFF}| = 12,\ |\varphi_{Ton}| = 4,\ |\varphi_{ON}| = 14,\ |\varphi_{Toff}| = 2,\ |\varphi_{Idle}| = 0$).

In the following, first of all, the complexity of the problems with a regular trend for energy

prices are investigated. Then, we studied the problem when the preemption of the jobs is allowed and finally, the problem with the same processing times for the jobs is addressed.

3.5.1 $1, c_t = c|states|TEC$

Here, the complexity of the problem with a constant energy prices is investigated.

Theorem 2 ***If the energy price during the horizon time is constant*** *($c_t = c; \forall t = 1, ..., T$)**, the problem*** *($1, c_t = c|states|TEC$)* ***is polynomial.***

Proof. In the problem $(1, c_t = c|states|TEC)$, the price of energy during all the periods is constant $(c_t = c\ ; \forall t = 1, \cdots, T)$, so, for any feasible solution, the expression of the objective function, denoted by Z_1, may be deduced from Equation (3.20) as:

$$\begin{aligned} Z_1 = &[(e_{OFF} * |\varphi_{OFF}|) + (e_{Ton} * |\varphi_{Ton}|) + \\ &(e_{ON} * |\varphi_{ON}|) + (e_{Toff} * |\varphi_{Toff}|) + (e_{Idle} * |\varphi_{Idle}|)] * c \end{aligned} \tag{3.26}$$

Let consider the solution $sol_1^\star$ such that: $|\varphi_{OFF}^{1^\star}| = T - (\beta_1 + P + \beta_2), |\varphi_{Ton}^{1^\star}| = \beta_1, |\varphi_{ON}^{1^\star}| = P, |\varphi_{Toff}^{1^\star}| = \beta_2, |\varphi_{Idle}^{1^\star}| = 0$, with the objective function value of $Z_1^\star$. For any other feasible solution of this problem (sol_1^i) with objective value as Z_1^i, the relation between Z_1^i and $Z_1^\star$ is as follow:

$$\begin{aligned} Z_1^i - Z_1^\star = &[(|\varphi_{OFF}^i| - |\varphi_{OFF}^{1^\star}|) * e_{OFF} + (|\varphi_{Ton}^i| - |\varphi_{Ton}^{1^\star}|) * e_{Ton} + \\ &(|\varphi_{Toff}^i| - |\varphi_{Toff}^{1^\star}|) * e_{Toff} + (|\varphi_{Idle}^i| - |\varphi_{Idle}^{1^\star}|) * e_{Idle}] * c \end{aligned} \tag{3.27}$$

Regarding to equations (2.1), (2.2), and (2.3), we have $e_{Ton} = e_{OFF} + \delta_1$, $e_{Toff} = e_{OFF} + \delta_2$, $e_{Idle} = e_{OFF} + \delta_3$, $e_{ON} = e_{OFF} + \delta_4$ with $(\delta_1, \delta_2, \delta_3, \delta_4 > 0)$, so:

$$\begin{aligned} Z_1^i - Z_1^\star &= [(|\varphi_{OFF}^i| + |\varphi_{Ton}^i| + |\varphi_{Toff}^i| + |\varphi_{Idle}^i| - T + P) * e_{OFF} \\ &+ (|\varphi_{Ton}^i| - \beta_1) * \delta_1 + (|\varphi_{Toff}^i| - \beta_2) * \delta_2 + (|\varphi_{Idle}^i| - 0) * \delta_3] * c \\ &= [(|\varphi_{Ton}^i| - \beta_1) * \delta_1 + (|\varphi_{Toff}^i| - \beta_2) * \delta_2 + |\varphi_{Idle}^i| * \delta_3] * c \end{aligned} \tag{3.28}$$

Based on equation 3.23 and the fact that $\lambda \geq 1$, we have $|\varphi_{Ton}^i| - \beta_1 \geq 0$, $|\varphi_{Toff}^i| - \beta_2 \geq 0$, $|\varphi_{Idle}^i| \geq 0$. Therefore, it can be concluded that $Z_1^i - Z_1^\star \geq 0$ which means that $Z_1^\star$, is a lower bound of this problem $(1, c_t = c|states|TEC)$. Since $sol_1^\star$ is also a feasible solution,

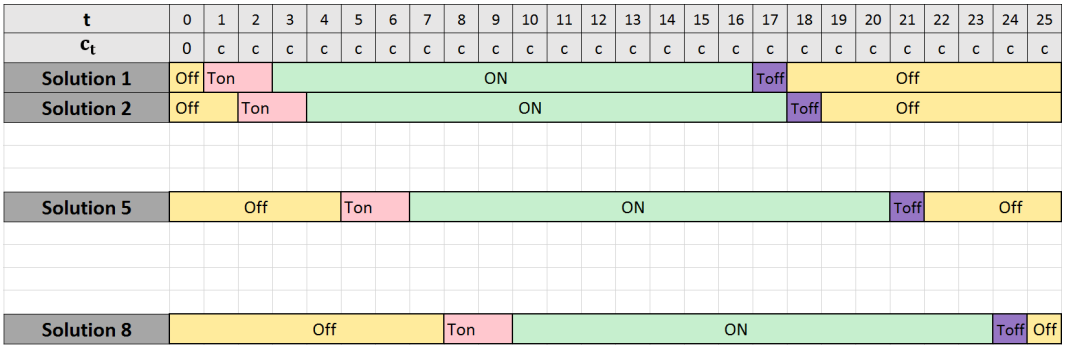

Figure 3.5 – The possible optimal solutions of $1, c_t = c|states|TEC$ for an example with $T = 25, P = 14, \beta_1 = 2, \beta_2 = 1$

therefore, $sol_1^\star$ is the optimal solution of this problem. Note that, $sol_1^\star$ is not a unique optimal solution of this problem,. All the feasible solutions which have the same value as $|\varphi_s^{1^\star}|$ for state $s \in \{OFF,\ Ton,\ ON,\ Toff,\ Idle\}$, have the same objective value and can be considered as the optimal solution.

Since in this problem there is not any priority between the jobs, consequently, the optimal solution of problem $(1, c_t = c|states|TEC)$ is when the machine has just one Ton and one Toff states, and processes all the jobs continuously (in any order). Besides, for the rest of the periods the machine is in OFF states and it has not any Idle state during the horizon. For example, for the presented problem in Fig. 3.5, there exist 8 different solutions with the same objective value. Any of these solutions can be considered as the optimal solution. Therefore, since the set of optimal solutions for this problem can be obtained directly, $(1, c_t = c|states|TEC)$ is polynomial. ■

3.5.2 $1, c_t < c_{t+1}|states|TEC$

In this section, we are interested to analyze the complexity of the same problem with the increasing energy prices of the periods.

Theorem 3 ***If the energy prices are increasing between two consecutive periods*** $(c_t < c_{t+1}; \forall t = 1, \cdots, T-1)$ ***and*** $e_{OFF} = 0$***, the problem*** $(1, c_t < c_{t+1}|states|TEC)$ ***is polynomial.***

Proof. Total energy consumption costs minimization of a production system can be reached by two ways: energy consumptions minimization and/or total energy costs minimization. In this problem $(1, c_t < c_{t+1}|states|TEC)$, unlike the previous one $(1, c_t = c|states|TEC)$, the energy

costs are different in each period, so, both of these ways may be used. For this purpose, the total energy consumptions of the machine should be minimized by allocating the minimum number of periods for each state (as it is demonstrated for $1, c_t = c|states|TEC$), and the total energy costs may be minimize by allocating the states during the low cost periods.

Let consider the solution $sol_2^\star$ as:

$$\begin{cases} \varphi_{Ton}^{2^\star} = \{1, \cdots, \beta_1\} \\ \varphi_{ON}^{2^\star} = \{\beta_1 + 1, \cdots, \beta_1 + P\} \\ \varphi_{Toff}^{2^\star} = \{\beta_1 + P + 1, \cdots, \beta_1 + P + \beta_2\} \\ \varphi_{OFF}^{2^\star} = \{\beta_1 + P + \beta_2 + 1, \cdots, T\} \\ |\varphi_{Idle}^{2^\star}| = 0 \end{cases} \tag{3.29}$$

with the objective function value of $Z_2^\star$ which may be computed from equation (3.20). For any other feasible solution of this problem (sol_2^i) with Z_2^i as objective value, the relation between Z_2^i and $Z_2^\star$ is as follow:

$$\begin{aligned} Z_2^i - Z_2^\star = e_{OFF} * (\textstyle\sum_{t\in\varphi_{OFF}^i} c_t - \sum_{t\in\varphi_{OFF}^{2^\star}} c_t) + e_{Ton} * (\sum_{t\in\varphi_{Ton}^i} c_t - \sum_{t\in\varphi_{Ton}^{2^\star}} c_t) + \\ e_{ON} * (\textstyle\sum_{t\in\varphi_{ON}^i} c_t - \sum_{t\in\varphi_{ON}^{2^\star}} c_t) + e_{Idle} * \sum_{t\in\varphi_{Idle}^i} c_t + e_{Toff} * (\sum_{t\in\varphi_{Toff}^i} c_t - \sum_{t\in\varphi_{Toff}^{2^\star}} c_t) \end{aligned} \tag{3.30}$$

The other possible solutions for this problem can be divided in two main sets. The first set of the solutions can be obtained by adding some non-processing states (Idle or middle-off) between two processing states. The second set can be obtained by changing the starting time of processing, through adding some initial-off states. All the other solutions may be obtained from a combination of these two cases.

For the first case, regarding equations (2.1), (2.2), and (2.3), obviously adding some non-processing states which consume more than OFF state ($e_{OFF} = 0$), causes an increase of the total energy consumptions and consequently the total energy consumption costs.

Let consider a general example (Fig. 3.6) such that: $1 < t_1 < t_2 < t_3 < t_4 < T$. If between two ON states, the machine goes to Idle state during period $t_2 + 1$, based on the equation (3.30), we have:

$$\begin{aligned} &Z_2^i - Z_2^\star = e_{Idle} * c_{t_2+1} + e_{ON} * (\sum_{t=t_2+2}^{t_3+1} c_t - \sum_{t=t_2+1}^{t_3} c_t) + \\ &+e_{Toff} * (\sum_{t=t_3+2}^{t_4+1} c_t - \sum_{t=t_3+1}^{t_4} c_t) + e_{OFF} * (\sum_{t=t_4+2}^{T} c_t - \sum_{t=t_4+1}^{T} c_t) \end{aligned} \tag{3.31}$$

Therefore,

$$\begin{aligned} &Z_2^i - Z_2^\star = e_{Idle} * c_{t_2+1} + e_{ON} * (c_{t_3+1} - c_{t_2+1}) + \\ &+e_{Toff} * (c_{t_4+1} - c_{t_3+1}) - e_{OFF} * c_{t_4+2} \end{aligned} \tag{3.32}$$

Since in the problem $1, c_t < c_{t+1}|states|TEC$, $e_{Idle} > e_{OFF} = 0$, and $\forall t' > t;\ c_{t'} > c_t$, we have:

$$Z_2^i - Z_2^\star \geq 0 \tag{3.33}$$

For the second case, let consider a general example (Fig. 3.7) such that: $1 < t_1 < t_2 < t_3 < T$ and $t_i < t'_i \quad \forall i = 1, 2, 3$. Based on the equation (3.30), we have:

$$\begin{aligned} &Z_2^i - Z_2^\star = e_{OFF} * (\sum_{t=1}^{t'_0} c_t + \sum_{t=t'_3+1}^{T} c_t - \sum_{t=t_3}^{T} c_t) + e_{Ton} * (\sum_{t=t'_0+1}^{t'_1} c_t - \sum_{t=1}^{t_1} c_t) + \\ &e_{ON} * (\sum_{t=t'_1+1}^{t'_2} c_t - \sum_{t=t_1+1}^{t_2} c_t) + e_{Toff} * (\sum_{t=t'_2+1}^{t'_3} c_t - \sum_{t=t_2+1}^{t_3} c_t) \end{aligned} \tag{3.34}$$

So,

$$\begin{aligned} &Z_2^i - Z_2^\star = e_{OFF} * (\sum_{t=1}^{t'_0} c_t - \sum_{t=t_3+1}^{t'_3} c_t) + e_{Ton} * (\sum_{t=t'_0+1}^{t'_1} c_t - \sum_{t=1}^{t_1} c_t) + \\ &e_{ON} * (\sum_{t=t'_1+1}^{t'_2} c_t - \sum_{t=t_1+1}^{t_2} c_t) + e_{Toff} * (\sum_{t=t'_2+1}^{t'_3} c_t - \sum_{t=t_2+1}^{t_3} c_t) \end{aligned} \tag{3.35}$$

Regarding to equations (2.1), (2.2), and (2.3), since in this problem, $e_{OFF} = 0$ and $\forall\, t' > t;\ c_{t'} > c_t$, we have:

$$Z_2^i - Z_2^\star \geq 0 \tag{3.36}$$

Thus, for any feasible solution as sol_2^i, we have $Z_2^i - Z_2^\star \geq 0$. It means that $Z_2^\star$, is a feasible lower bound of this problem $(1, c_t < c_{t+1}|states|TEC)$, and $sol_2^\star$ is the optimal solution.

So, in the optimal solution, the machine must be in Ton state from period 1 to β_1. Then, the

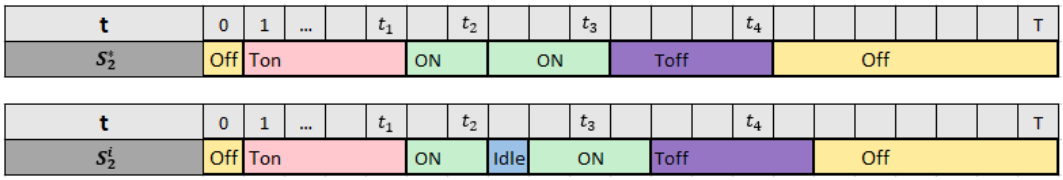

Figure 3.6 – The comparison between solution sol_2^i and $sol_2^\star$ of problem $1, c_t < c_{t+1}|states|TEC$: case1

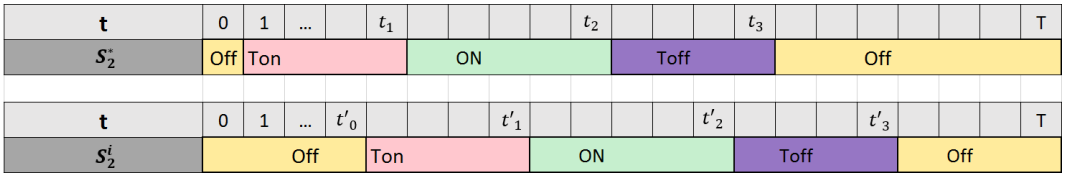

Figure 3.7 – The comparison between solution sol_2^i and $sol_2^\star$ of problem $1, c_t < c_{t+1}|states|TEC$: case2

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
c_t	0	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Solution 1	Off	Ton	ON															Toff	Off							

Figure 3.8 – The optimal solution for an instance of problem with $c_t < c_{t+1} \quad (\forall t = 1, \cdots, T-1)$

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
c_t	0	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
Solution 8	Off								Ton	ON															Toff	Off

Figure 3.9 – The optimal solutions for an instance of problem with $c_t > c_{t+1} \quad (\forall t = 1, \cdots, T-1)$

jobs must be processed from period $\beta_1 + 1$ to $\beta_1 + 1 + P$ in any order, and finally the machine must be in Toff and OFF states consecutively (Fig. 3.8).

Therefore, $1, c_t < c_{t+1}|states|TEC$ is a polynomial problem, since the optimal solution can be obtained in polynomial time with the value of $Z_2^\star$. ■

Note that, with the same approach but in the backward way, it can be proved that the problem $1, c_t > c_{t+1}|states|TEC$ is also polynomial (Fig. 3.9).

3.5.3 $1, TOU|states, pmtn|TEC$

After analysing the affect of energy costs on the complexity of this problem, in this section, we are interested to study the complexity of the preemptive case. For this purpose, the same approach as the presented dynamic programming approach in chapter 2 is used.

Graph construction steps

As it is described before (section 2.4.1), a finite graph which consists of $T+1$ levels and different number of nodes in each level is considered to model this problem. In this graph the levels represent the periods, and the nodes represent the status of the system regarding to the performed jobs in each period.

To construct the related graph for each instance of the problem, three steps must be performed.

Step 1: putting the nodes As it is explained in details at section 2.4.1, in this approach, there are $x+1$ possibilities for every node $k \in \{1, ..., P\}$. Note that, for distinction between the initial and final states of the machine and the other states, one node I and one node F are presented to indicate the initial and final OFF states respectively. In the other words, node I indicates that the machine did not start to perform the jobs yet, while, node F indicates that the machine already performed all the jobs completely. Besides, the node $k \in [1, P]$ represents that k units of the processing jobs are performed among P units. Therefore, the total number of nodes for the related graph is equal to:

$$|V| = P \times (x+1) + 2 \cong TP$$

Step 2: drawing the edges The edges of the graph can be divided in three main sets. The first one indicates the Idle states or the initial and final OFF states, between two side by side nodes with the same number. The second one illustrates the initial Ton, processing the next job, and final Toff states. The third one shows the middle-OFF stats between two ON states. The procedure of drawing the edges is as the following:

After selecting one level to turn on the machine, an edge with the length of $(\beta_1 + 1)$ between I and 1 must be drawn that contains one Ton state and one ON state for processing the first unit of the jobs. Then, 3 output edges are possible for the node 1: one with the next node 1 that indicates Idle state, one with node 2 by length of 1 that means processing the next unit of the jobs, and the third one with node 2 by different lengths to represent middle-OFF states during some periods.

So, x edges are required to connect $x+1$ possible nodes for each $k \in \{I, 1, ..., P, F\}$, that present

the first set of edges. Total number of these edges are equal to $(P+2) \times x$.

Moreover, $(x+1)$ edges are needed to exhibit different transition states between I and 1 for Ton, k and $k+1$ for processing all the jobs ($k \in [1, P-1]$), and between P and F for Toff states. Total number of the second set of the edges is equal to $(P+1) \times (x+1)$.

Since the machine should be in OFF state for at least one period during the middle-OFF states, the required number of edges to consider all the possibilities for middle-OFF between k in level l' and node $k+1$ in level l ($l' \in \{l_{min(k)}, \cdots, x+k-\beta_2-1\}$) is equal to:

$$\sum_{i=1}^{x-(\beta_1+\beta_2)} i = \frac{(x-(\beta_1+\beta_2)) \times (x-(\beta_1+\beta_2)+1)}{2} \quad ;\forall k \in [1, P-1]$$

Consequently, the total number of the edges for this graph is equal to:

$$|E| = ((P+2) \times x) + ((P+1) \times (x+1)) + ((P-1) \times [\frac{(x-(\beta_1+\beta_2)) \times (x-(\beta_1+\beta_2)+1)}{2}])$$

$$\Rightarrow \quad |E| \cong T^2 P$$

In this approach, the difference between the fixed sequence version of this study which is presented in chapter 2 and it's preemption version is in number of the edges. For the fixed sequence version, the non-processing states can be allocated between two processing jobs, it means that the related edges for the middle-off and Idle states may appear between two jobs just when the first job is completed and the second one is not start. For the preemption version, these edges can appear at each moment between two jobs (see figure 2.7 and 3.10. So, the related graph for the preemption version has more edges than the related graph for the fixed-sequence version, whereas, both of them have the same number of nodes.

As example, the related graph for the considered instance with $P = 5, T = 15, \beta_1 = 2, \beta_2 = 1, x = 6$ is presented at Fig. 3.10. This graph is composed of $|V| = 7 \times 7 = 49$ nodes and $|E| = (7 \times 6) + (6 \times 7) + (4 \times [\frac{3 \times 4}{2}]) = 108$ edges.

Recurrence relationship formulation

Since the objective function of this problem is the minimization of total energy costs when energy prices are different in each period, it is necessary to enter the energy cost of each transition in

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
c_t	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

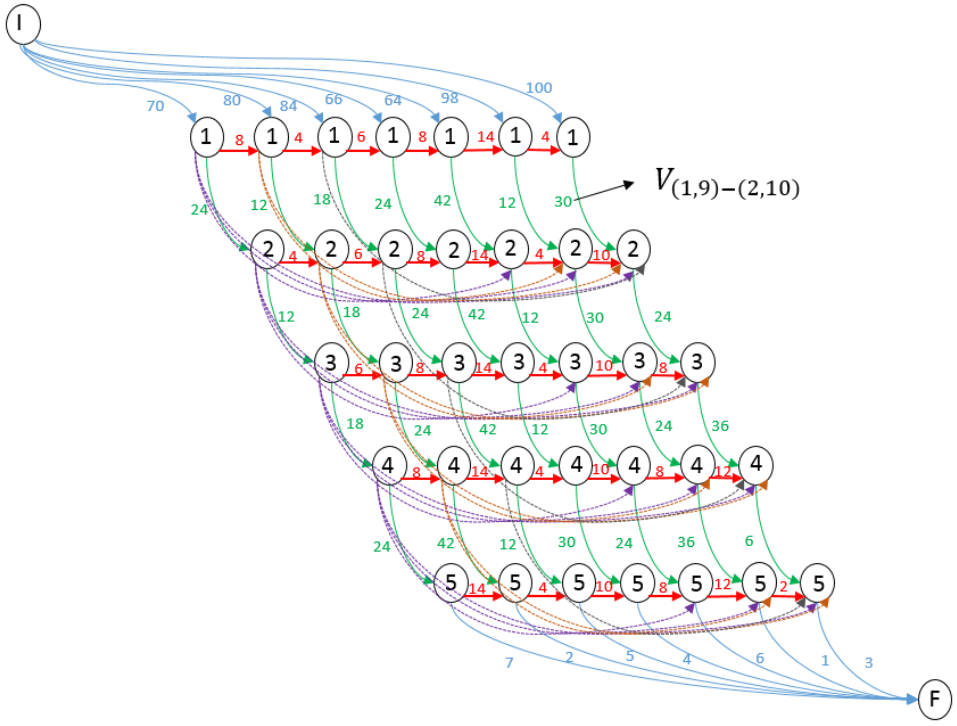

Figure 3.10 – Graph for preemption problem

the presented graph. For this purpose, the energy cost of each transition considered as the value of the edge that connect every two nodes. After that, the graph is complete and the shortest path that starts from node I in level 0 and ends at node F in level T, is the best solution of the problem. To find the shortest path of a graph in a single source problem, Dijkstra's algorithm is the most famous and efficient algorithm when all the edge values are positives. Generally, this algorithm use for finding the shortest path between the source node and every other nodes, but, it can be used for finding the shortest path from a single source node to a single destination node (like our problem) by stopping when the shortest path to the destination node is determined. The Pseudocode of this algorithm is presented in Table 3.2.

As it is presented in section 2.4.3, the complexity of this algorithm for the presented dynamic programming approach is equal to:

$$O(|E| + |V| \log |V|) = O(T^2P + TP \log TP) = O(T^2P + TP \log T + TP \log P) \tag{3.37}$$

Dijkstra's algorithm

Consider 0 as the source node and $(P+1)(x+1)-1$ as the destination node.
Shortest path to source vertex is zero (spt[0]=0).
Shortest path to other vertex is infinity ($spt[v] = \infty \quad ; \forall v \in [1, (P+1)(x+1)-1]$).
M= adjacency matrix of graph V
for $\forall i, j \in V$
{
if $spt[i] + M[i][j] < spt[j]$
$spt[j] = spt[i] + M[i][j]$
}
spt[(P+1)(x+1)-1]= the shortest path value for arriving to the destination vertex.

Table 3.2 – Dijkstra's algorithm pseudocode

Therefore, the final complexity of this problem can be simplified as follow:

$$\begin{cases} T > P, T > \log T \\ (\forall T, P > 0) \rightarrow T^2 P > TP \log T > TP \log P \\ O(T^2 P + TP \log TP) \cong O(T^2 P) \end{cases} \tag{3.38}$$

Since, in the presented problem, the biggest value for P is T, so, all these results approve that in the worst case, this problem is a polynomial of degree 3 or a cubic polynomial.

3.5.4 $1, TOU|states, p_j = p|TEC$

The last case analysis of the sub-problems of $(1, TOU|states|TEC)$ concerns the problems with the same processing times for the jobs. The complexity of this problem is studied in the following.

Theorem 4 ***If the jobs have the same processing times*** $(p_j = p \quad ; \forall j = 1, \cdots, n)$***, the problem*** $(1, TOU|states, p_j = p|TEC)$ ***is polynomial.***

Proof. To schedule the states of the machine during a horizon time, two important factors are energy consumption of the machine during each state, and the unit of energy price in each period. In the general problem $(1, TOU|states|TEC)$, it is considered that the energy consumption of the machine during the ON states is independent from the processed job. So, the only parameter which caused a preference between the jobs is their processing times. Thus, when the jobs have the

same processing times, the machine consumes the same amount of energies for performing them. Let us consider that, each job needs p periods to be processed ($p_j = p$), and the machine consumes e_{ON} unites of energy per period during the ON states. So, the machine consumes $e_{ON} * p$ units of energy to process any job. Regarding to this information, there is not any advantage between the different sequence of the jobs $\{(e_{ON} * p) - (e_{ON} * p) - (e_{ON} * p) - \cdots - (e_{ON} * p)\}$. It means that, when the machine is in the state ON, any job may be processed without affecting the optimality of the solution. So, the problem $1, TOU|states, p_j = p|TEC$, can be considered as a fixed sequence problem in which each job needs p unit times as process times ($1, TOU|states, sequence, p_j = p|TEC$). The general version of this problem ($1, TOU|states, sequence|TEC$) is previously examined in section 2.4.1, and proved to be polynomial. Therefore, the considered problem in this section ($1, TOU|states, p_j = p|TEC$) which is it's sub-problem, is also a polynomial problem. ■

As it is proved in this section, all the studied sub-problems are polynomial while, the general problem ($1, TOU|states|TEC$) is an NP-hard problem. Therefore, in the following section it is attempted to propose some efficient resolution methods for this problem to obtain the near optimal solutions for any size of the problem.

3.6 Lower bounds For $(1, TOU|states|TEC)$ problem

A usual tool to evaluate the performances of the approximate methods for an NP-hard problem is to obtain some lower bounds. For this reason, in the following, we attempt to propose some lower bounds for the problem $1, TOU|states|TEC$.

From the given set of the period's energy cost $C = \{c_t \ ; \forall t = 1, \cdots, T\}$, let consider the set $\tilde{C} = \{\tilde{c}_1, \tilde{c}_2, \cdots, \tilde{c}_T\}$, which contains the period's energy cost in the increasing order, such that $\tilde{c}_1 \leq \tilde{c}_2 \leq \cdots \leq \tilde{c}_T$. Then, the following relation can also be written:

$$\sum_{t=1}^{\theta} \tilde{c}_t \leq \sum_{t=1}^{\theta} c_t \quad \forall \theta = 1, \cdots, T \tag{3.39}$$

Regarding equations (2.1), (2.2), and (2.3), the OFF state has the minimum energy consumption between all the non-processing states. Therefore, in the cases that the unite energy prices are increasing, obviously adding some non-processing states which consume more than OFF state

would increase the total energy consumptions and consequently the total energy consumption costs. That is why, in this section, the minimum number of required periods for Ton and Toff states are considered for defining the lower bounds, and states of the machine during all the remaining periods are considered as OFF state.

3.6.1 First lower bound (LB_1)

Let define LB_1 as the cost of allocating the cheapest periods to each state. So, we have:

$$LB_1 = (e_{OFF} \times \sum_{t=1}^{T-(\beta_1+P+\beta_2)} \tilde{c}_t) + (e_{Ton} \times \sum_{t=1}^{\beta_1} \tilde{c}_t) + (e_{ON} \times \sum_{t=1}^{P} \tilde{c}_t) + (e_{Toff} \times \sum_{t=1}^{\beta_2} \tilde{c}_t) \quad (3.40)$$

Lemma 5 *LB_1 is a lower bound of $(1, TOU|states|TEC)$.*

Proof. Regarding the equation (3.20), the optimal value of the total energy costs ($Z^\star$) for this problem can be computed as:

$$\begin{aligned} Z^\star = {} & e_{OFF} \times \textstyle\sum_{t\in\varphi^{3\star}_{OFF}} c_t + e_{Ton} \times \sum_{t\in\varphi^{3\star}_{Ton}} c_t + \\ & e_{ON} \times \textstyle\sum_{t\in\varphi^{3\star}_{ON}} c_t + e_{Toff} \times \sum_{t\in\varphi^{3\star}_{Toff}} c_t + e_{Idle} \times \sum_{t\in\varphi^{3\star}_{Idle}} c_t \end{aligned} \quad (3.41)$$

Based on the problem's assumption (equations (3.23),(3.24),(3.25)), the cardinal of $\varphi^\star_s$ for each state in the optimal solution are as follows ($\lambda^\star \geq 1$):

$$\begin{cases} |\varphi^\star_{Ton}| = \lambda^\star \times \beta_1; \quad |\varphi^\star_{ON}| = P; \\ \\ |\varphi^\star_{Toff}| = \lambda^\star \times \beta_2; \quad |\varphi^\star_{Idle}| \geq 0; \quad |\varphi^\star_{OFF}| \geq \lambda^\star \end{cases} \quad (3.42)$$

$$|\varphi^\star_{Ton}| + |\varphi^\star_{Toff}| + |\varphi^\star_{Idle}| + |\varphi^\star_{OFF}| = T - |\varphi^\star_{ON}| = T - P \quad (3.43)$$

Based on equation 3.39, we have the following equations:

$$\begin{cases} \sum_{t=1}^{\beta_1} \tilde{c}_t \leq \sum_{t=1}^{|\varphi^\star_{Ton}|} \tilde{c}_t \leq \sum_{t\in\varphi^\star_{Ton}} c_t \\ \sum_{t=1}^{P} \tilde{c}_t \leq \sum_{t\in\varphi^\star_{ON}} c_t \\ \sum_{t=1}^{\beta_2} \tilde{c}_t \leq \sum_{t=1}^{|\varphi^\star_{Toff}|} \tilde{c}_t \leq \sum_{t\in\varphi^\star_{Toff}} c_t \\ \sum_{t=1}^{T-(\beta_1+P+\beta_2)} \tilde{c}_t \leq \sum_{t=1}^{|\varphi^\star_{OFF}|} \tilde{c}_t \leq \sum_{t\in\varphi^\star_{OFF}} c_t \end{cases} \tag{3.44}$$

Regarding to the above equations (3.44), the following relation can be obtained for LB_1 and $Z^\star$:

$$LB_1 - Z^\star \leq 0 \tag{3.45}$$

Therefore, LB_1 is a lower bound for the problem $1, TOU|states|TEC$. ■

3.6.2 Second lower bound (LB_2)

To define the first lower bound of this problem (LB_1), the non-preemption and precedence constraints for the states of the machine, and the fact that the machine must be in one and only one state per period are relaxed. Whereas high importance is given to the unit of energy price in each period. For this reason, LB_1 value may does not correspond to a feasible solution, since more than one state can assume for the same period, the jobs be processed preemptively and a Toff state be located before a ON state. In reality, if the machine starts to process job j in period t, the machine must be in ON state from period t to $t + p_j - 1$, and it is not possible to be in other state during these periods or perform another job. For this purpose, we define the second lower bound (LB_2) which is more near to the reality. It sorts the periods based on their energy costs and allocates them to Ton, ON, $Toff$, and OFF states, respectively and continuously. By this way, the machine has only one state per period, but the precedence constraints for the states and the non-preemption constraints are not considered yet. Because, it sorts the periods only on

the basis of their energy price, not their sequence. Therefore, LB_2 is computed as follow:

$$\begin{aligned} LB_2 = (e_{Ton} \times \textstyle\sum_{t=1}^{\beta_1} \tilde{c}_t) + (e_{ON} \times \textstyle\sum_{t=\beta_1+1}^{\beta_1+P} \tilde{c}_t) + \\ (e_{Toff} \times \textstyle\sum_{t=\beta_1+P+1}^{\beta_1+P+\beta_2} \tilde{c}_t) + (e_{OFF} \times \textstyle\sum_{t=\beta_1+P+\beta_2+1}^{T} \tilde{c}_t) \end{aligned} \tag{3.46}$$

Lemma 6 *LB_2 is a lower bound of $(1, TOU|states|TEC)$.*

Proof. The problem $1, TOU|states|TEC$ with $\tilde{c}_t$ $(\forall t = 1, \cdots, T)$, is equivalent to the problem with the increasing energy prices $(1, c_t < c_{t+1}|states|TEC)$ whom optimal solution is provided in section II. So, the optimal solution of problem $1, c_t < c_{t+1}|states|TEC$ may be used as a lower bound of the problem $1, TOU|states|TEC$. ■

3.6.3 Third lower bound (LB_3)

Let consider $\underline{C}_j$ that computes the minimum cost of performing job j $(\forall\ j = 1, ..., n)$ non-preemptively during its possible periods. As it is explained before, the possible periods that the machine can be in Ton, Toff and ON states depend to the total number of periods, the number of extra periods, and the number of required periods for performing each job. So, $\underline{C}_j$ may be formulated as follow.

$$\begin{aligned} &\underline{C}_j = \min\{c_t + c_{t+1} + \cdots + c_{t+p_j-1}\}; \\ &\forall\ t \in \{\beta_1 + 1, \cdots, T - p_j - \beta_2\};\ \forall\ j \in \{1, \cdots, n\} \end{aligned} \tag{3.47}$$

Then, following the same idea, the minimum costs for Ton, Toff and OFF states are obtained with the following formulations.

$$\underline{C}_{Ton} = \min\{c_t + c_{t+1} + \cdots + c_{t+\beta_1-1}\};\ \forall\ t \in \{1, 2, \cdots, x+1\} \tag{3.48}$$

$$\underline{C}_{Toff} = \min\{c_t + c_{t+1} + \cdots + c_{t+\beta_2-1}\};\ \forall\ t \in \{T - x - \beta_2, \cdots, T - \beta_2\} \tag{3.49}$$

$$\underline{C}_{OFF} = \sum_{t=1}^{T-(\beta_1+P+\beta_2)} \tilde{c}_t \tag{3.50}$$

As it has been discussed before, regarding equations (2.1), (2.2), and (2.3), the Idle state consumes more than OFF state, so, all the remaining non-processing states are considered as OFF states. The idea of the third lower bound, named LB_3, is to allocate to each state its minimum costs $\underline{C}_j$ $(\forall j \in \{1, \cdots, n\})$, $\underline{C}_{Ton}$, $\underline{C}_{Toff}$, $\underline{C}_{OFF}$:

$$LB_3 = (e_{OFF} \times \underline{C}_{OFF}) + (e_{Ton} \times \underline{C}_{Ton}) + (e_{ON} \times \sum_{j=1}^{n} \underline{C}_j) + (e_{Toff} \times \underline{C}_{Toff}) \tag{3.51}$$

Lemma 7 *LB_3 is a lower bound of $1, TOU|states|TEC$.*

Proof. To evaluate $\underline{C}_j; \forall j \in \{1, \cdots, n\}$, the constraint that the machine can process one job per period is relaxed and the processing order for the jobs is not considered. Moreover, to evaluate $\underline{C}_j$ $(\forall j \in \{1, \cdots, n\})$, $\underline{C}_{Ton}$, $\underline{C}_{Toff}$, $\underline{C}_{OFF}$, the constraints that the machine must be in just one state per period, and the relationship between different state of the machine are relaxed. So, for a feasible solution of $1, TOU|states|TEC$ we have the following relations:

$$\underline{C}_{OFF} \leq \sum_{t \in \varphi^{\star}_{OFF}} c_t; \quad \underline{C}_{Ton} \leq \sum_{t \in \varphi^{\star}_{Ton}} c_t; \quad \underline{C}_{Toff} \leq \sum_{t \in \varphi^{\star}_{Toff}} c_t; \quad \sum_{j=1}^{n} \underline{C}_j \leq \sum_{t \in \varphi^{\star}_{ON}} c_t \tag{3.52}$$

And we have:

$$\begin{cases} e_k \times \underline{C}_k \quad \leq e_k \times \sum_{t \in \varphi^{\star}_k} c_t; \ \forall \ k \in \{OFF, Ton, Toff\} \\ \\ e_{ON} \times \sum_{j=1}^{n} \underline{C}_j \quad \leq e_{ON} \times \sum_{t \in \varphi^{\star}_{ON}} c_t \end{cases} \tag{3.53}$$

Consequently:

$$LB_3 \quad \leq Z^{\star} \tag{3.54}$$

Therefore, LB_3 is a lower bound of this problem. ■

3.6.4 Fourth lower bound (LB_4)

Moreover, the optimal solution of the preemption version of this problem can be defined as LB_4.

Lemma 8 *LB_4 is a lower bound of $1, TOU|states|TEC$.*

States and transitions	Power consumption	required periods
ON	4 kW	$\sum$ process times
OFF	0 kW	-
Idle	2 kW	-
Toff	1 kW	1
Ton	5 kW	2

Table 3.3 – Energy consumption profile of a machine. ([61])

Proof. As it is demonstrated in the previous section (section II), the preemption version of this problem $(1, TOU|states, pmtn|TEC)$ which is a subproblem of $1, TOU|states|TEC$, is polynomial. So, it's optimal solution may be used as the fourth lower bound (LB_4) of this problem. ■

3.6.5 Numerical evaluation for the proposed lower bounds

To evaluate the efficiency of the proposed lower bounds in this study, several randomly generated instances are considered. Based on the presented examples in a pervious study ([61]), the machine setup data for all the examined instances in this study are identical and considered as Table. 3.3. For each size of the problem, several instances have been examined. To generate the instances, the unit of energy price in each period, as well as the processing times of the jobs are randomly generated between $[1, 10]$ and $[1, 5]$, respectively. In Table 3.4, the gaps between the objective value of each lower bound and the obtained optimal solution by CPLEX software are given in percentage. These results are presented for the problems smaller than $(35, 209)$, because CPLEX software was not able to find the optimal solution for the larger instances during 3 hours or 10800 seconds limitation time. The numerical results have been illustrated by minimum, average and maximum obtained gap value for each problem size. The results show that between LB_1, LB_2, and LB_3, in all the cases LB_2 proposed a better bound. Among these lower bounds, LB_4 which is the obtained optimal solution of the preemptive case of this problem by CPLEX, finds the closest solution to the optimal solution. The ranking order for these lower bounds is as follows:

$$Gap_{LB_4} < Gap_{LB_2} < Gap_{LB_1} < Gap_{LB_3}$$

Moreover, an analysis of the variance (ANOVA) with a confidence level of 95% was taken using the Minitab.17 software to check the statistical validity of the results (Fig. 5.6). As can be seen

(n,T)		Gap_{LB_1}	Gap_{LB_2}	Gap_{LB_3}	Gap_{LB_4}
(5,30)	Min	19.41	8.14	18.94	0.00
	Average	**39.79**	**27.70**	**39.65**	**0.07**
	Max	54.75	40.68	56.00	0.37
(10,50)	Min	20.15	13.31	42.53	0.00
	Average	**33.66**	**26.44**	**50.27**	**2.90**
	Max	50.35	43.94	62.13	11.59
(15,70)	Min	20.02	14.51	47.44	0.00
	Average	**28.76**	**23.52**	**57.10**	**0.00**
	Max	35.60	30.88	68.39	0.00
(20,90)	Min	15.52	11.05	53.20	0.00
	Average	**29.83**	**25.43**	**57.38**	**0.00**
	Max	36.03	31.23	62.37	0.00
(25,110)	Min	25.96	22.44	44.55	0.00
	Average	**30.29**	**26.59**	**56.68**	**0.00**
	Max	34.27	30.65	70.64	0.00
(30,130)	Min	20.29	17.10	54.55	0.00
	Average	**25.45**	**22.51**	**60.67**	**0.00**
	Max	28.29	25.98	71.42	0.00
(35,209)	Min	24.34	22.57	48.42	0.00
	Average	**27.27**	**25.48**	**53.58**	**3.11**
	Max	30.68	28.91	61.46	5.17
Average		**30.72**	**25.38**	**53.62**	**0.87**

Table 3.4 – The comparison results between the proposed lower bounds and obtained optimal solutions by CPLEX in percentage

in this figure, for each problem size the interval of the gaps for all the proposed lower bounds ($LB_1,\ LB_2,\ LB_3,\ LB_4$) are presented. In all the cases, LB_4 has the minimum interval of the gaps.

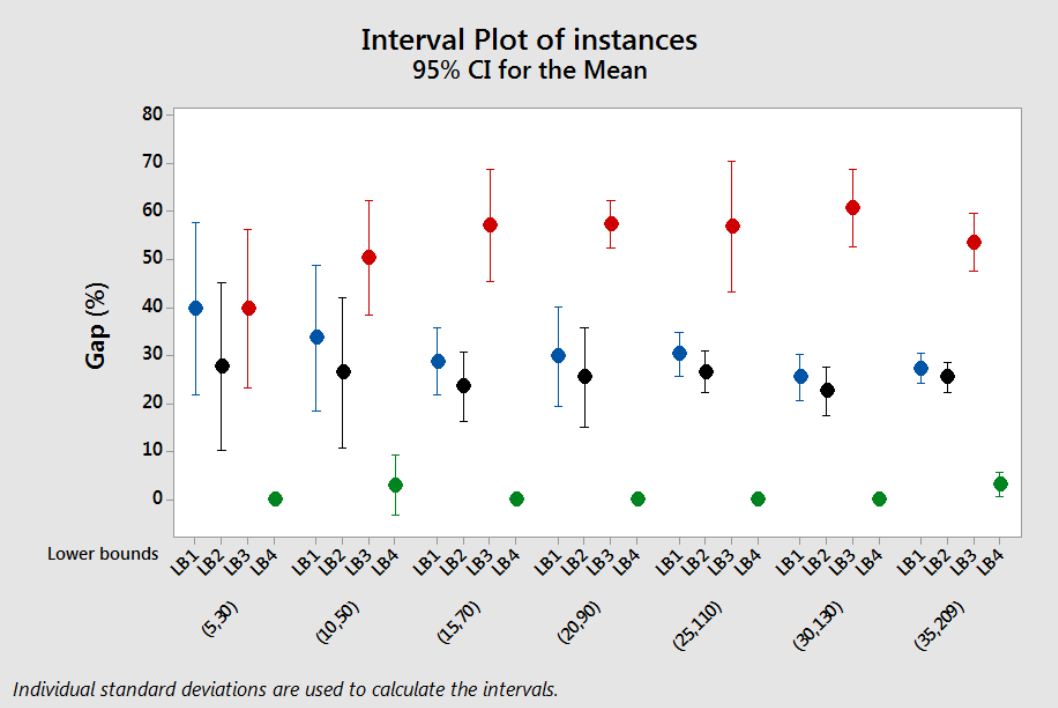

Figure 3.11 – Performance comparison of the lower bounds with the obtained optimal solutions by CPLEX

3.7 Optimization methods for $(1, TOU|states|TEC)$ problem

The exact methods can not find the optimal solution for the large size instances of an NP-hard problem in a reasonable time. So, for the presented problem in this chapter which is NP-hard $(1, TOU|states|TEC)$, we attempted to propose some heuristic and meta-heuristic algorithms. For this purpose, a heuristic algorithm and a genetic algorithm are presented to optimize the machine's state and jobs' production scheduling simultaneously. The description of these two algorithms and their comparison are presented in the following subsections.

3.7.1 Heuristic method

One of the most popular methods for solving an NP-hard problem is to develop a heuristic algorithm which is able to find an efficient feasible solution (near optimal) for large size instances of problem. So, first of all, we attempted to present an intelligent and effective heuristic algorithm for our problem. As it is mentioned in the previous chapter, the basic condition to have the

feasible solutions for a problem with a fixed number of periods is that, the number of existing periods must be greater than or equal to the number of required periods. These required periods are for switching on the machine, performing all the jobs completely and then switching it off. The difference between these two numbers indicates the number of extra periods which can be calculate by equation 5.1 and indicate by x. The principles of the presented heuristic in this chapter are based on allocating the non-processing states (Ton, OFF, Idle and Toff) to the machine during these extra periods. The extra periods may be placed anywhere on a production shift (at the beginning, end or middle).

Heuristic's procedures

The proposed algorithm is divided into two general steps (forward and backward steps) to specify the machine's state period by period and consider all the possibilities for non-processing states.

In each step, the algorithm investigates $(x+1)$ different situations (from 0 to x) for turning on the machine at the beginning or turning it off at the end of the horizon, and selects the minimum objective function among $(x+1)$ different solutions. By this way, for each solution, some extra periods will be allocated to OFF state at the first stage and then, the algorithm will decide for the rest (among OFF or Idle states) in other periods. Finally, the solution is the one with the minimum objective value between the best solutions of forward and backward steps. Thus, to obtain the best solution, $2*(x+1)$ different solutions must be considered and compared with each other. This method (determine the number of checking solutions) causes reducing dependency of the algorithm to the number of jobs and periods (size of the problem) and increases its efficiency.

For the forward step, in the first solution, the OFF state assigns to the machine at period zero. In the second solution, the OFF state assigns to the machine at period zero and one. In the third solution, the OFF state assigns to the machine at period zero, one and two. By the same way, for the next solutions, each time the OFF state assigns to the machine for one more periods. So, finally in the last solution ($(x+1)th$) of this step, the OFF state assigns to the machine at $x+1$ periods ($\forall t \in [0, x]$). The possible solutions in the forward step for an example with 3 jobs and 14 periods, are given in figure 3.12.

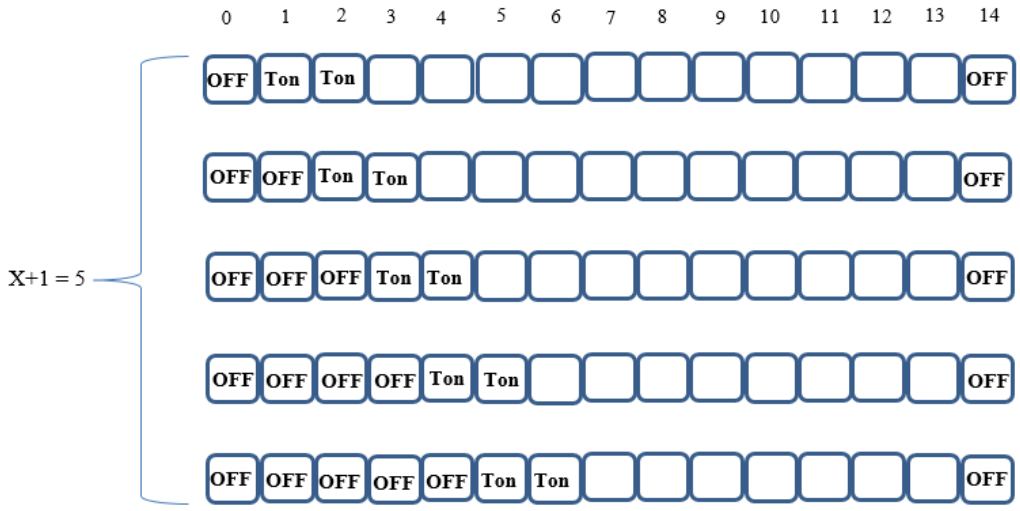

Figure 3.12 – Forward step's solutions for instance (3,14)

In the backward step, the same procedures as the forward step will be performed, but by allocating the OFF states to the machine at the end of horizon. It means that, the algorithm assigns the OFF state to the machine in the last period for the first solution, and in period T and $T-1$ for the second one. For the other solutions, the OFF state assigns to the machine from one earlier period. Therefore, in the $(x+1)th$ solution the machine will be in the OFF states for periods $t \in [T-x, T]$. The possible solutions in the backward step for an example with 3 jobs and 14 periods, are given in figure 3.13.

To complete each solution, just after turning on the machine (in forward step) or before turning of the machine (in backward step), the algorithm selects the best job to process based on the minimum average value of their energy consumption cost in the related period. As it is shown in figure 3.14, in the first solution of forward step, the second job ($j2$) is selected. Also, as can be seen in figure 3.15, in the second solution of backward step, the first job ($j1$) is selected.
After processing the chosen job, the algorithm will select the best state for the machine based on the minimum average value of their energy consumption cost. In the first solution of forward step for our instance (figure 3.14), Idle state has been chosen for period 4.
These steps will be continued to process all of the jobs completely and specify the machine's

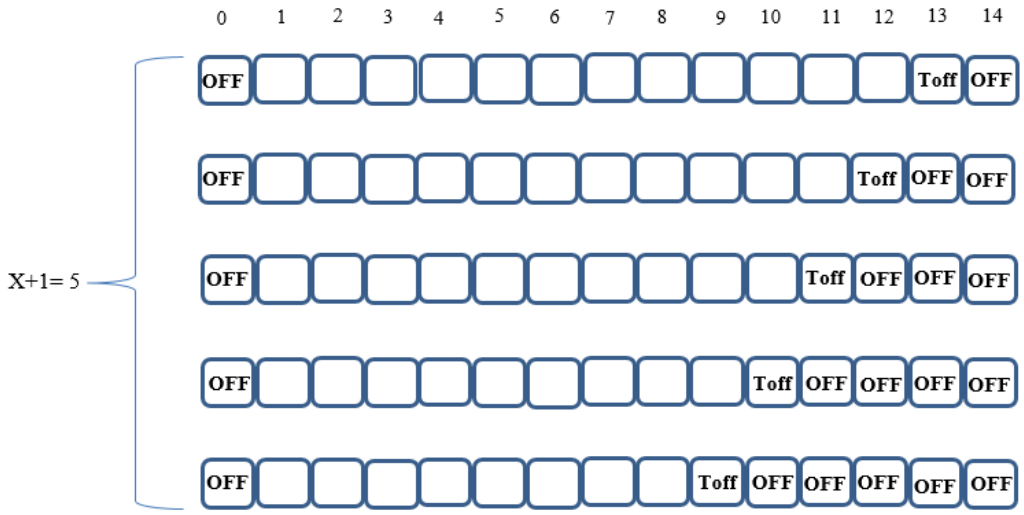

Figure 3.13 – Backward step's solutions for instance (3,14)

Period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Energy cost	0	4	2	3	4	4	5	2	2	4	9	8	5	3	8
Job	1	2	3												
Process time	3	1	2												

Table 3.5 – Energy costs in each period & Process time of each job

state during all the periods. After that, the algorithm's process is finished and it computes the objective value of each solution.

For better understanding of this algorithm's procedure, it is applied on a small example with 3 jobs and 14 periods. The energy cost of each period and processing time of each job are detailed in Table 3.5. According to these parameters values we have $x = 4$, so, 5 different solutions must be compared in each step. These solutions and their objectives' value (total energy cost) have been presented in figures 3.14 and 3.15. As can be seen, the best solution of the forward step is the last one with the objective value of 168, while, among the backward's solutions, the best solution has the objective value of 114. Therefore, the best scheduling plan found by the proposed heuristic algorithm for this problem is the last solution of backward step with total energy cost= 114. The pseudo code of this algorithm is represented in Table 3.6. In the following section, a genetic algorithm based on our heuristic's method is also presented to examine the possibility of improving the obtained solutions by this algorithm.

Heuristic Algorithm : HA

if $(x < 0)$ **then** the problem is infeasible.
else if $(x = 0)$ **then**
just one possibility for machine states' position and the jobs can be processed in different sequences with the same objective values.
else
Step 1: for $(i = 0$ to $x)$
1. put the machine in OFF state for periods $p \in [0, i]$ and T.
2. turn on the machine just after period i.
3. once the machine is in processing state, process the job with minimum mean of energy cost requirement for processing in this period.
4. choose the machine's state with minimum mean of energy cost requirement among {processing, Idle or OFF (if $x - i > 3$)}.
if processing state is selected **then** go back to the stage 3.
else if Idle state is selected **then**
put the machine in Idle, consider $x - i = x - i - 1$ and return to the stage 4.
else OFF state is chosen **then**
turn off the machine for the $(x - i)$ next periods, (consider Toff, OFF and Ton steps), put $x - i = 0$ and return to the stage 4.
end if
5. **if** $(x = 0)$ **then** process the remaining jobs based on the numerical order.
end if
6. compute the objective function for each solution (Z_1)
end for
let Z_{min_1}=$min\{Z_1\}$
Step 2: for $i = 0$ to x
1. put the machine in OFF state for periods $p \in [T - i, T]$ and 0.
2. turn on the machine just for the period before $(T - i - 1)$.
3. do the same as step 1 in backward way from final period (stages 4 and 5).
4. compute the objective function for each solution (Z_2).
end for
let Z_{min_2}=$min\{Z_2\}$
let Z_{min}=$min\{Z_1, Z_2\}$ (minimum objective value of all the solutions)
end if

Table 3.6 – The heuristic algorithm

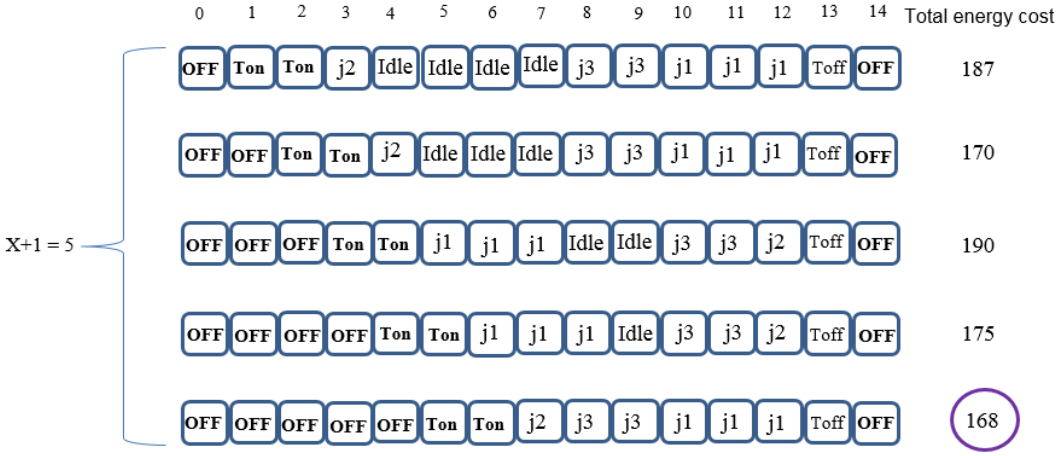

Figure 3.14 – Forward step of the heuristic algorithm

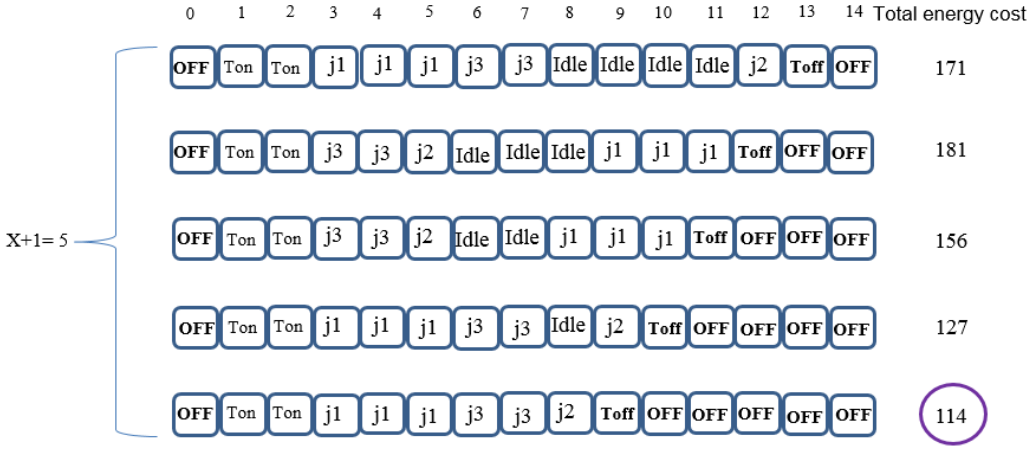

Figure 3.15 – Backward step of the heuristic algorithm

3.7.2 Genetic algorithm

Meta-heuristics techniques which are usually inspired by the biological process of natural selection can be considered to identify a valid solving procedure. Genetic Algorithms are one of these techniques ([78]) to find good solutions for complex combinatorial optimisation problems. As well as other evolutionary approaches, like simulated annealing and tabu search, it has been applied to scheduling and sequencing problems earlier with success ([79]). The implementation of a genetic algorithm (figure 3.16) needs to establish several concepts such as chromosome, initial population, parent selection method, crossover and mutation operators. These concepts for our problem are

defined in the following sections.

Initial population and chromosome representation

Each solution of this problem consists of T periods for which the machine's state must be defined. So, in this genetic algorithm, each chromosome is represented by $T + 1$ genes and each gene identifies the machine's state in any period.

To distinguish between the different states, the numbers of 1, 2, 4 and 5 are assumed to represent OFF, Ton, Toff and Idle states, respectively. Besides, an integer number more than 10 ($w > 10$), represents that the machine is in ON state and processes the $(w - 10)th$ job. The related chromosome for the presented solution at Figure 3.2 is shown in Figure 5.5. Since in this instance the number of periods is 32, so, this chromosome consists of 33 genes. The number 11 in $15th$ gene means that during period 14 the machine processes the job $11 - 10 = 1$. Also, the number 5 in $19th$ gene means that during period 18 the machine is in Idle state.

Usually, the genetic algorithm (GA) starts with a randomly generated initial population. The proposed GA uses an initial population which will be generated based on our proposed heuristic algorithm with a population size of 300. Thus, for each individual, first of all the period for turning on the machine must be selected randomly. Then, the job's number to perform will be chosen randomly. After completing the job, the machine's state must be selected among ON, Toff and Idle states arbitrarily. These procedures must be continued to process all of the jobs until the last period. Finally, the objective value of the solution will be computed as a fitness function to classify the generated chromosome's quality.

Operating parameters

For completing the GA's procedure it is necessary to choose three operators for parents selection, crossover and mutation procedures. For this purpose, we use the same operators as the existing study in the literature ([61]). Therefore, the roulette wheel operation is considered for parents selection. Moreover, single point and swap method are considered as crossover and mutation operators, respectively, to produce the new children.

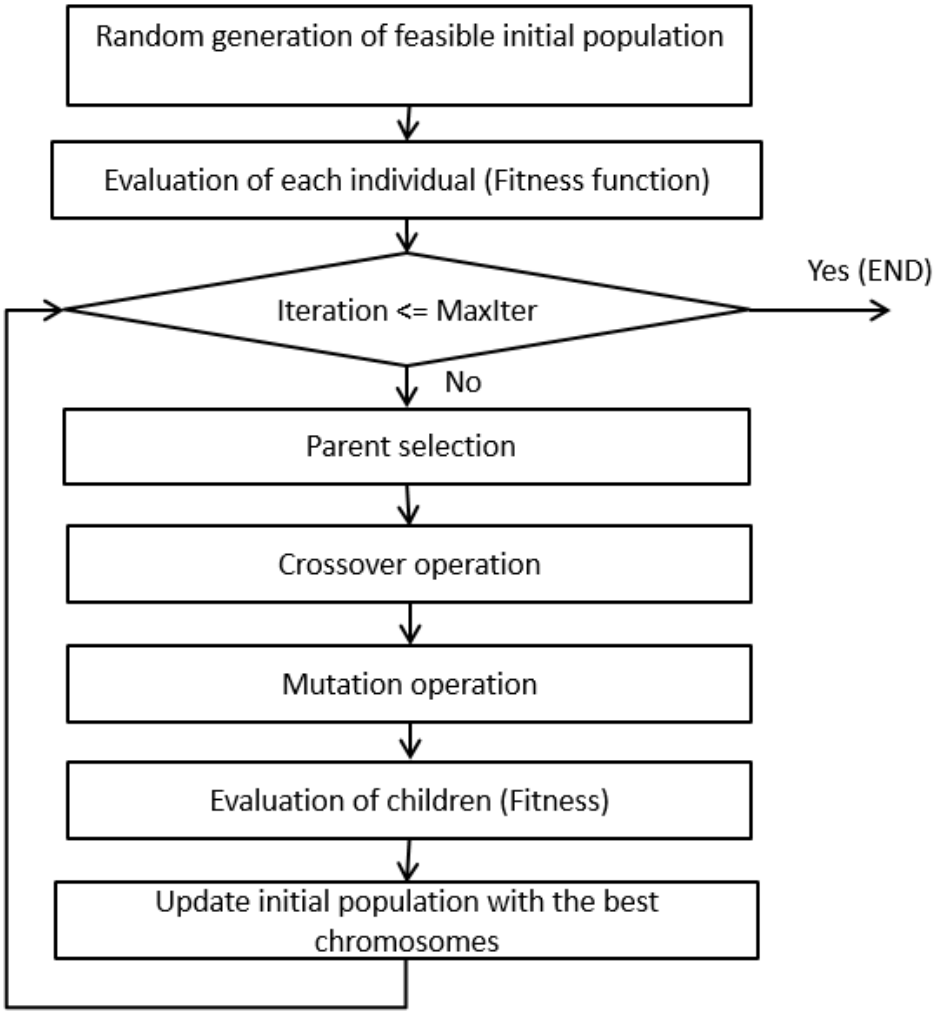

Figure 3.16 – Genetic algorithm's procedure

1	1	1	1	1	1	1	1	2	2	12	12	14	14	11	11	11	5	5	15	15	15	13	13	13	13	4	1	1	1	1	1	1

Figure 3.17 – The related chromosome for the obtained solution in Figure 3.2

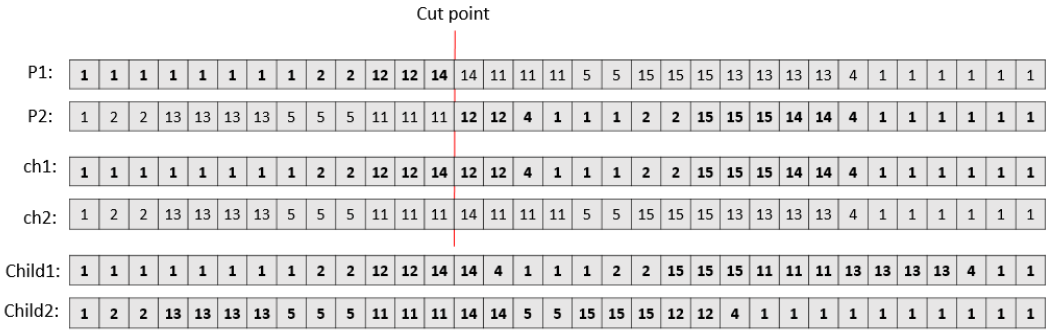

Figure 3.18 – Single point crossover operation

Crossover operation For the single point crossover operation, the cut point will be chosen randomly from the period's index. The first offspring will be composed of the first parent (from the beginning to the cut point) and the second parent (from the first gene after the cut point to the end of the chromosome). The second offspring will be obtained by the same way and reverse the parents.

After producing the children, a correction procedure must be done to convert the not feasible solutions to the feasible one. One example of these operations are illustrated in Figure 3.18.

Mutation operation The next step in the genetic algorithm is to apply the mutation operator. Swap method is selected based on the literature. For this purpose, a chromosome from the initial population will be randomly selected and the mutation will be performed on the randomly selected gene by changing its value. Then, like the crossover operating, it is necessary to check the feasibility of the obtained offsprings and correct them if they are not feasible.

One example of these operations are illustrated in Figure 3.19.

Update the initial population The final step in the first iteration of GA is to update the population, because the crossover and mutation procedures will increase the population size. So, for the next iteration, the best chromosomes among all the initial population and the collected

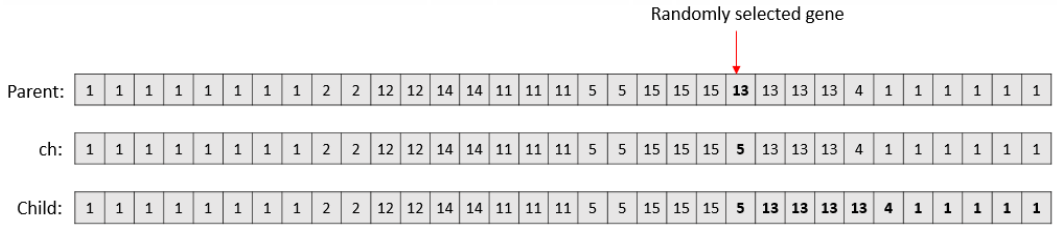

Figure 3.19 – Mutation operation: swap method

population in the current iteration, must be selected based on the value of their fitness function as much as the size of the population. These procedures must be accomplished for a maximum iteration number which is considered as 300 times in our proposed GA.

3.7.3 Numerical experiments

The proposed algorithms have been implemented in $C++$ Visual Studio 2015. In order to evaluate their performances, several numerical instances have been generated. The generation of these instances is inspired from the literature. The computational times was limited to 3 hours or 10800 seconds and the numerical results have been illustrated by minimum, average and maximum value of obtained results for each criterion in Tables 3.7 and 3.8.

For the small size instances, the gap between the objective values and the computation times of these algorithms and the exact method were collected to illustrate the efficiency of these algorithms. In table 3.7, Gap_{HA} and Gap_{GA}, illustrate the obtained gap between HA-exact method and GA-exact method's objective values in percentage. Also, CPU_{Cplex}, CPU_{HA} and CPU_{GA} represent the computation time of exact method, HA and GA in second. Since the exact method is not able to find any solution during three hours for the problems with more than 35 jobs and 209 periods, a comparison between HA and GA was performed for large size problems that has been presented in Table 3.8.

In Shrouf et al.'s work ([61]), they did not investigate the efficiency of their proposed genetic algorithm for the problems larger than 60 jobs and 135 periods, while, in this chapter, we studied the efficiency of our proposed algorithms for the problems as large as (200,1200).

The obtained results demonstrate that our proposed algorithms provide the optimal solutions in some examples and nearly optimal solutions for others in few seconds (gap of 2.2% is achieved in average by HA and 1.82% by GA in comparing to the objective value of exact solution for small size instances). Also, among all the computational experiments, the biggest gap of 5.67% and 7.31% are obtained by the HA and GA respectively. These results demonstrate great accuracy and efficiency of both the proposed algorithms. Generally, the GA presents a better solution in contrast to HA, but with more time-consuming. The minimum and maximum computing time are (0.36 s, 8.44 s) and (18.53 s, 424.28 s) for the HA and GA respectively (the average computing time for the HA is 1.52 seconds and for the GA is equal to 119.13 seconds). These information illustrate that impact of the problem's size on HA's performance is lower than GA. Moreover, an analysis of the variance (ANOVA) with a confidence level of 95% was taken using the Minitab.17 software to check the statistical validity of the results (Figures 3.20 and 3.21). As can be seen, the minimum and maximum variation range of gaps belong to GA for the instances with problem size of (5-30) and (30-130) respectively. Also, just in these two sets of examples, the heuristic algorithm provides the better solutions generally. In large size instances, the variation range of gaps between HA and GA are less than 3% and in average are approximately equal to 1%.

(n, T)		Gap_{HA} (%)	Gap_{GA} (%)	CPU_{Cplex} (s)	CPU_{HA} (s)	CPU_{GA} (s)
(5, 30)	Min	0.00	1.10	0.25	0.71	20.06
	Average	**1.06**	**1.58**	**0.7**	**1.32**	**31.68**
	Max	2.28	2.28	1.34	2.74	39.15
(10, 50)	Min	0.00	0.00	1.15	0.53	32.91
	Average	**1.5**	**0.83**	**3.79**	**0.81**	**40.44**
	Max	3.76	2.42	7.09	1.31	47.16
(15, 70)	Min	0.35	0.00	7.34	0.51	18.53
	Average	**2.18**	**0.9**	**36.27**	**0.8**	**32.92**
	Max	4.69	2.44	115.15	1.23	41.8
(20, 90)	Min	0.00	0.86	6.89	0.6	22.5
	Average	**1.81**	**1.67**	**46.64**	**0.76**	**44.17**
	Max	3.45	2.49	169.96	1.06	60.61
(25, 110)	Min	0.46	0.46	31.87	0.37	28.47
	Average	**2.3**	**1.56**	**284.01**	**0.61**	**46.32**
	Max	3.89	2.59	959.53	0.95	60.88
(30, 130)	Min	1.55	0.00	331.8	0.51	32.92
	Average	**3.56**	**3.68**	**609.64**	**0.71**	**56.04**
	Max	5.67	7.31	901.65	1.4	103.37
(35, 209)	Min	1.54	1.2	2536.09	0.36	74.92
	Average	**3.02**	**2.58**	**9147.22**	**0.93**	**97.53**
	Max	4.94	4.94	10800	1.92	105.07

Table 3.7 – Comparative results among Heuristic, Genetic Algorithm and exact method

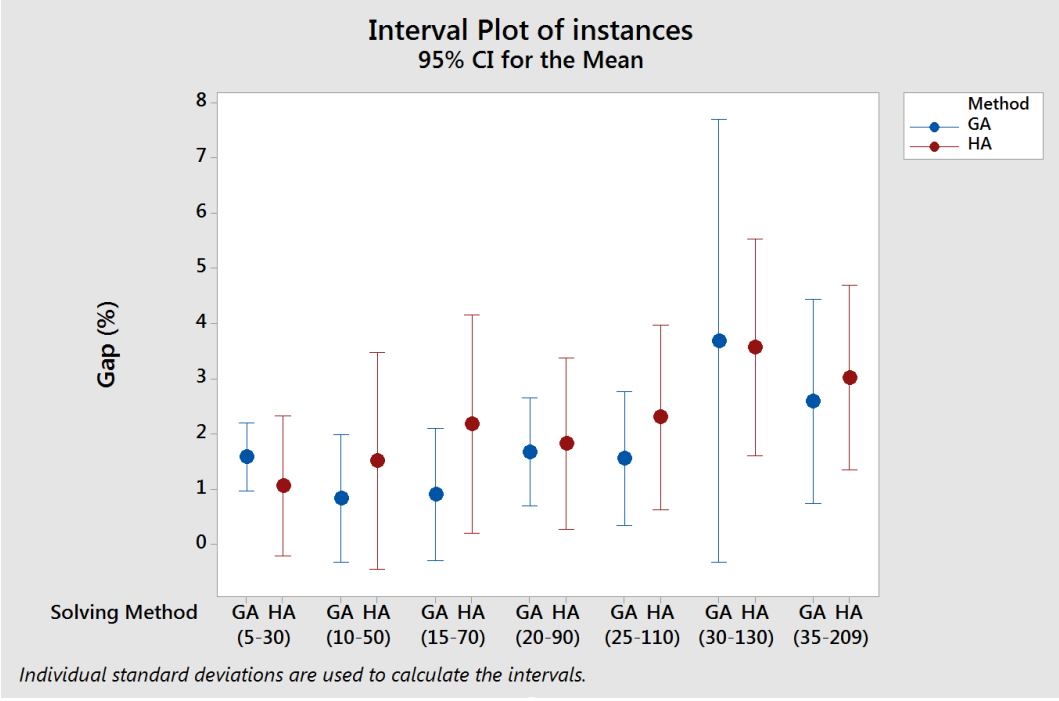

Figure 3.20 – Performance comparison of GA and HA with Exact method

(n, T)		*Gap* (%)	CPU_{HA} (s)	CPU_{GA} (s)
$(40, 249)$	Min	0.09	0.65	98.57
	Average	**0.71**	**0.85**	**110.99**
	Max	1.1	1.31	123.66
$(45, 270)$	Min	0.28	0.79	72.54
	Average	**1.42**	**0.91**	**101.99**
	Max	2.99	1.07	121.87
$(50, 300)$	Min	0.25	0.76	63.97
	Average	**0.86**	**0.94**	**78.32**
	Max	1.75	1.31	100.36
$(60, 360)$	Min	0.12	0.98	102.72
	Average	**1.46**	**1.1**	**125.16**
	Max	2.13	1.23	145.83
$(70, 420)$	Min	0.32	1.04	122.68
	Average	**1.53**	**1.14**	**146.69**
	Max	2.6	1.2	174.15
$(80, 480)$	Min	0.22	1.09	111.32
	Average	**0.73**	**1.17**	**136.69**
	Max	1.5	1.24	205.47
$(90, 540)$	Min	0.06	1.29	80.25
	Average	**0.51**	**1.6**	**111.44**
	Max	1.63	2.07	156.9
$(100, 600)$	Min	0.25	1.42	139.08
	Average	**0.46**	**1.96**	**155.85**
	Max	0.59	2.26	190.35
$(150, 900)$	Min	0.04	2.1	297.34
	Average	**0.61**	**3.33**	**363.96**
	Max	1.69	4.32	424.28
$(200, 1200)$	Min	0.09	4.94	270.86
	Average	**0.46**	**6.99**	**345.64**
	Max	1.2	8.44	404.67

Table 3.8 – Comparative results of Heuristic and Genetic Algorithm for medium and large size instances

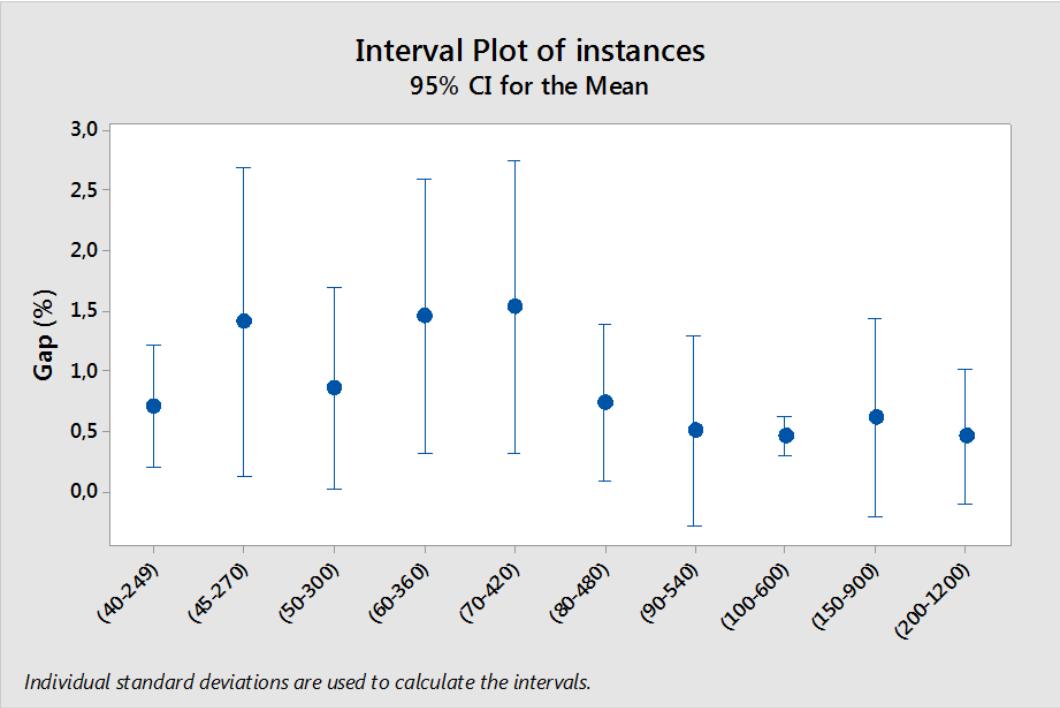

Figure 3.21 – Performance comparison of GA against HA for large size problems

3.8 Conclusion

Several multi-states single machine scheduling problems, when the jobs consume the same units of energy and the machine has only one processing speed are addressed in this chapter. It is proved that when the energy price during the time horizon is constant, increasing or decreasing, this problem is polynomial. Moreover, when the jobs can be process preemptively, as well as when the jobs have the same processing time, the problem is polynomial too. The general version of this problem $(1, TOU|states|TEC)$ which attempt to find the optimal sequence for processing the jobs non-preemptively and the optimal state of the machine is each period, is NP-hard. Therefore, a heuristic algorithm and a genetic algorithm are presented for this problem to find the near optimal solutions for all size of the problem rapidly. Moreover, several lower bounds are proposed for this problem to evaluate the efficiency of these algorithms.

In the next chapter we will deal with a multi-states single machine scheduling problem when the jobs consumes different amount of energy. For this purpose, two versions of this problem with an uniform speed and a speed-scalable machine are studied.

4

Multi-states and multi-speeds single-machine energy-efficient scheduling problem

Outline of the current chapter

4.1 Introduction

In the previous chapters of this thesis, the scheduling problems with the same energy consumption of the jobs are addressed. In reality, for some manufacturing industries, processing different jobs with the same machine need different energy consumptions. It may come from the preformed jobs or from different processing speeds option. Therefore, in this chapter, we study the more general version of the problem with multi-states and multi-speeds for a single machine. Besides, the complexity of several sub problems are also analyzed.

The remainder of this chapter is organized into seven sub-sections. In section 4.2, the definition of the problem with its assumptions and constraints are presented. In section 4.3, two mathematical models are presented for the considered problem, and the comparison between them is represented. In section 4.5, a heuristic algorithm and a genetic algorithm, as well as a memetic algorithm are presented to solve the problem. In section 4.4, the complexity of several sub problems are studied. Finally, the brief conclusion of this chapter is presented in section 4.6.

4.2 Problem presentation

This section deals with the scheduling problem of several jobs on a multi-states and multi-speeds single machine $(1, TOU|states, speeds, q_j|TEC)$. So, we keep all the main assumptions and constraints of chapter 2. Moreover, for the studied problem in this section, the energy consumption of the machine during state ON depends on two factors: the performed job and the processing speed of the machine. So, there are different possibilities for processing time of each job and the consumed energy value by the machine. That means for each job $j = 1, ..., n$ with v_j possible speeds, there are different values for the processing time as $P_j = \{p_{j,1}, \cdots, p_{j,v_j}\}$, and for each $p_{j,i}$, a corresponding energy consumption $q_{j,i}$ is associated. $Q_j = \{q_{j,1}, \cdots, q_{j,v_j}\}$ is the set of the different energy consumptions of job $j = 1, ..., n$. Based on the fact that, processing a job more faster takes less times and consumes more units of energy, the following relations are

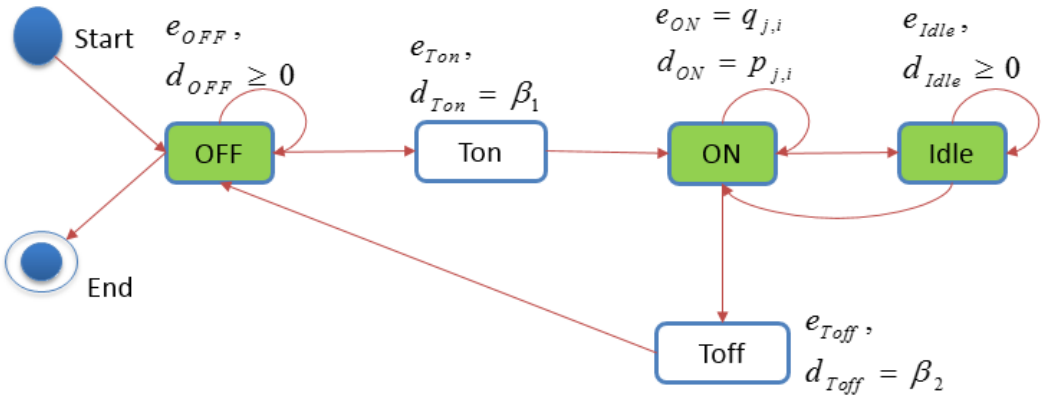

Figure 4.1 – Machine states and possible transitions

considered:

$$p_{j,1} > p_{j,2} > \cdots > p_{j,v_j} \quad ;\forall j \in \{1, \cdots, n\} \tag{4.1}$$

$$q_{j,1} < q_{j,2} < \cdots < q_{j,v_j} \quad ;\forall j \in \{1, \cdots, n\} \tag{4.2}$$

The objective of this problem is to find the most economical production schedule in terms of energy consumption costs during the whole time horizon. The machine states and the possible transitions as well as the energy consumptions of each one are illustrated in Figures 5.7.

4.2.1 Illustrative example

Let us consider a scheduling problem of 5 jobs in 32 periods on a machine with 3 speed levels for each job. A feasible solution of this instance is presented in Figures 4.2. As can be seen, the first job is processed with speed 1, the second and the fourth jobs are performed by speed 3, and the third and the fifth jobs are processed with speed 2.

In this kinds of problem, the idea is to process the job with a higher speed when the unit of energy cost is low and reverse.

Parameter	Value	Parameter	Value	Parameter	Value
$p_{1,1}$	3	$p_{1,2}$	2	$p_{1,3}$	1
$p_{2,1}$	5	$p_{2,2}$	4	$p_{2,3}$	2
$p_{3,1}$	6	$p_{3,2}$	4	$p_{3,3}$	3
$p_{4,1}$	5	$p_{4,2}$	3	$p_{4,3}$	2
$p_{5,1}$	4	$p_{5,2}$	3	$p_{5,3}$	2
$q_{1,1}$	4	$q_{1,2}$	5	$q_{1,3}$	6
$q_{2,1}$	3	$q_{2,2}$	4	$q_{2,3}$	5
$q_{3,1}$	2	$q_{3,2}$	4	$q_{3,3}$	5
$q_{4,1}$	1	$q_{4,2}$	2	$q_{4,3}$	3
$q_{5,1}$	2	$q_{5,2}$	3	$q_{5,3}$	5
β_1	2	β_2	1	e_{OFF}	0
e_{Idle}	2	e_{Ton}	5	e_{Toff}	1

Table 4.1 – The parameters' values for an instance with 5 jobs, 32 periods and 3 speeds

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	Cost
c_t	0	8	8	8	4	4	4	3	3	3	2	2	2	2	2	10	10	10	10	3	3	3	2	2	2	2	6	6	3	3	3	5	5	
ON							16	12	12	15	10	8	8	8	8							9	6	6	6	6								
OFF	0	0	0	0													0	0	0									0	0	0	0	0	0	
Idle																																		
Turn on					20	20														15	15												
Turn off																10											6							
The schedule	Off				Ton		J1-1			J2-3		J3-2				Toff	Off			Ton		J4-3		J5-2			Toff	Off						206

Figure 4.2 – A feasible solution for the problem with 5 jobs, 32 periods and 3 speeds

4.3 Mathematical formulation

As the first contribution, two new mathematical models are proposed in the following. To describe these programming models, first of all, the parameters and decision variables, as well as, the objective function and the constraints which are used in the first model are presented. Then, we describe the second model based on the first one and we define the additional parameters and decision variables.

4.3.1 First model

Parameters:

T : Total number of periods;

c_t: Unit of energy price in period t;

n : Number of jobs

v_j: Number of possible processing speed for job j;

$p_{j,i}$: Processing time of job j with speed $i = \{1, \cdots, v_j\}$ (in a number of periods);

$q_{j,i}$: Energy consumption of job j per period which is associated with $p_{j,i}$;

s : States of the machine (s={1,2,3} for ON, OFF and idle states, respectively.);

e_s: Energy consumption of the machine during state $s = \{1, 2, 3\}$;

$e_{ss'}$: Energy consumption of the machine in transiting between s and s' $(s, s' = \{1, 2\} | s \neq s')$;

$d_{ss'}$: Required number of periods for switching from state s to s' $(s \neq s')$;

Decision variables:

In this formulation, two binary decision variables are used to describe the state of the machine in each period:

$$\alpha_{s,t} = \begin{cases} 1 & \text{If the machine is in state } s \text{ during period } t = 0, ..., T \\ 0 & \text{Otherwise} \end{cases}$$

$$\beta_{ss',t} = \begin{cases} 1 & \text{If the machine is in transition from state } s \text{ to } s' \text{ in period } t = 0, ..., T \\ 0 & \text{Otherwise} \end{cases}$$

Moreover, two binary decision variables are also used to define the status of the jobs in each period:

$$x_{j,i} = \begin{cases} 1 & \text{If job } j \text{ is processed with speed } i = \{1, \cdots, v_j\} \\ 0 & \text{Otherwise} \end{cases}$$

$$y_{j,i,t} = \begin{cases} 1 & \text{If the machine processes job } j \text{ with speed } i \text{ in period } t = 0, ..., T \\ 0 & \text{Otherwise} \end{cases}$$

Mathematical formulation:

$$Min \sum_{t=0}^{T} c_t \left(\sum_{j=1}^{n} \sum_{i=1}^{v_j} q_{j,i} \cdot y_{j,i,t} + \sum_{s=2}^{3} E_s \cdot \alpha_{s,t} + \sum_{s=1}^{3} \sum_{s'=1}^{3} E_{ss'} \cdot \beta_{ss',t} \right) \tag{4.3}$$

$$\sum_{s=1}^{3}\alpha_{s,t}+\sum_{s=1}^{3}\sum_{s'=1}^{3}\beta_{ss',t}=1 \qquad ;\forall t\in\{0,\cdots,T\} \tag{4.4}$$

$$\alpha_{s,t}\leq \sum_{s'=1|d_{ss'}=0}^{3}\alpha_{s',t+1}+\sum_{s=1|d_{ss''}\geq 1}^{3}\sum_{s''=1}^{3}\beta_{ss'',t+1} \qquad ;\forall t\in\{0,\cdots,T-1\},\forall s\in\{1,2,3\} \tag{4.5}$$

$$\beta_{ss',t}\leq \beta_{ss',t+1}+\alpha_{s',t+1} \qquad ;\forall t\in\{0,\cdots,T-1\},\forall s,s'\in\{1,2,3\},d_{ss'}\geq 1 \tag{4.6}$$

$$\sum_{t'=t+1}^{t+d_{ss'}}\beta_{ss',t'}\geq(\alpha_{s,t}+\beta_{ss',t+1}-1)\cdot d_{ss'} \qquad ;\forall t\in\{0,\cdots,T-1\},\forall s,s'\in\{1,2,3\},d_{ss'}\geq 1 \tag{4.7}$$

$$\beta_{ss',t}+\beta_{ss',t+d_{ss'}}\leq 1 \qquad ;\forall t\in\{0,\cdots,T-t_{ss'}\},\forall s,s'\in\{1,2,3\},d_{ss'}\geq 1 \tag{4.8}$$

$$\sum_{j=1}^{n}\sum_{i=1}^{v_j}y_{j,i,t}=\alpha_{1,t} \qquad ;\forall t\in\{1,\cdots,T\} \tag{4.9}$$

$$\sum_{j=1}^{n}\sum_{i=1}^{v_j}y_{j,i,t}\leq 1 \qquad ;\forall t\in\{0,\cdots,T\} \tag{4.10}$$

$$\sum_{t'=0}^{t-p_{j,i}}y_{j,i,t'}+\sum_{t'=t+p_{j,i}}^{T}y_{j,i,t'}\leq p_{j,i}\cdot(1-y_{j,i,t}) \qquad ;\forall t\in\{p_{j,i},\cdots,T-p_{j,i}-1\},\forall j\in\{1,\cdots,n\},\forall i\in\{1,\cdots,v_j\} \tag{4.11}$$

$$\sum_{i=1}^{v_j}x_{j,i}=1 \qquad ;\forall j\in\{1,\cdots,n\} \tag{4.12}$$

$$\sum_{t=0}^{T}y_{j,i,t}\geq p_{j,i}\cdot x_{j,i} \qquad ;\forall j\in\{1,\cdots,n\},\forall i\in\{1,\cdots,v_j\} \tag{4.13}$$

$$\alpha_{2,t}=1 \qquad ;\forall t\in\{0,T\} \tag{4.14}$$

In this model, the objective value depends on the machine states, the processing job, the processing speed (energy consumption of the machine), and the unit of electricity price in each period (equation (4.3)). Equation (4.4) expresses that in each period the machine must be in one of the possible states (ON, OFF, Idle, Ton, and Toff). Equations (4.5) and (4.6) limit the machine's state in each period regarding to the machine's state in previous period. Equations (4.7) and (4.8) identify the required number of periods for Ton and Toff states. Equation (4.9) indicates that the machine may process at most one job per period, and if the machine processes job j during period t, it must be in ON state ($s=1$). Equation (4.10) imposes the constraint that the machine can process at most one job per period. Equation (4.11) translates the non-preemption constraints of the jobs. Equation (4.12) imposes the constraint that each job must be processed with only one speed. Equation (4.13) specifies the processing time of each job, regarding to its

processing speed. Equation (4.14) identifies that the machine is in OFF state during the initial and final periods.

4.3.2 Second model

In the first formulation, two decision variables ($y_{j,i,t}$ and $x_{j,i}$) and three indexes (j, i and t) are used to modelize the problem. For the second model, it is attempted to propose a more efficient model to formulate the problem by using just one decision variable and two indexes. This formulation is inspired by the approach proposed by [80].

Let us define N jobs such that $N = \sum_{j=1}^{n} v_j$. Let define $J = \{J_1, J_2, ..., J_n\}$ the set of all the job such that $J_1 = \{1, 2, ..., v_1\}$, $J_j = \{v_{j-1} + 1, ..., v_{j-1} + v_j\}$; $\forall j = 2, ..., n$.

Let also consider the set $P = \{p_1, ..., p_N\}$ of the processing time of each job $k \in J$ and the set $Q = \{q_1, ..., q_N\}$ of the energy consumptions of the jobs $k \in J$ such that q_k is the unit of energy consumption corresponding to the processing time p_k.

Let also define $R_{k,t} = \sum_{\tau=t}^{t+p_k-1} c_\tau$ that computes the sum of energy unit cost if the machine performs job $k = 1, ..., N$ from period t to period $t + p_k - 1$. Then, $q_k * R_{k,t}$ represents the total energy consumption for performing job k from period t.

Moreover, $x_{k,t}$ is used as a decision variable to formulate the problem:

$$x_{k,t} = \begin{cases} 1 & \text{If job } k \text{ begins to be processed at period } t \quad (k = 1, ..., N) \\ 0 & \text{Otherwise} \end{cases}$$

So, in this model, the objective function can be written as follow:

$$Min \sum_{t=0}^{T} \sum_{k=1}^{N} q_k \cdot x_{k,t} \cdot R_{k,t} + \sum_{t=0}^{T} c_t \cdot \left(\sum_{s=2}^{3} E_s \cdot \alpha_{s,t} + \sum_{s=1}^{3} \sum_{s'=1}^{3} E_{ss'} \cdot \beta_{ss',t} \right) \tag{4.15}$$

Also, the constraints in which variable $y_{j,i,t}$ and $x_{j,i}$ are present (equations: (4.9) to (4.13)) must be changed. Therefore, equation (4.16) is replaced (4.9), (4.11) and (4.13). Moreover, equations (4.17) and (4.18) respectively replace equations (4.10) and (4.12). The new equations are given in the following:

$$\sum_{s=2}^{3} \alpha_{s,t} + \sum_{s=1}^{3} \sum_{s'=1}^{3} \beta_{ss',t} = 1 - \sum_{k=1}^{N} \sum_{t'=t-p_k+1}^{t} x_{k,t'} \quad ; \forall t \in \{0, \cdots, T\} \tag{4.16}$$

$$\sum_{k=1}^{N} x_{k,t} \leq 1 \qquad ;\forall t \in \{0, \cdots, T\} \tag{4.17}$$

$$\sum_{k \in J_j} \sum_{t=0}^{T} x_{k,t} = 1 \qquad ;\forall j \in \{1, \cdots, n\} \tag{4.18}$$

It must be mention that, in the second model, all the related constraints for defining the machine's state in each period are kept as the same as in the first model (equations: (4.4), (4.5), (4.6), (4.7) and (4.8)).

Note that, the second model integrates a pre-treatment phase that computes the value of parameters $R_{k,t}$ for any k and t.

4.3.3 Comparison

The performance of these two mathematical models have been examined by several randomly generated instances. For this purpose, CPLEX 12.6.1 software is used to solve instances with the exact method (Branch and Cut). Five different examples are randomly generated for each instance by changing the processing times and the energy consumptions of the jobs among $[1, 8]$, as well as the unit of energy price in each period among $[1, 10]$. It must be mention that these generations are inspired from the literature [61]. The computation time for all the experiments was set to 1 hour or 3600 seconds. By using the first model, CPLEX was able to find the optimal solutions for problems smaller than 15 jobs, 5 speeds, and 120 periods. Therefore, the results of the models in terms of the number of constraints and variables, as well as the computation time are compared together. These results are presented in Table 4.2. The second model is faster than the first one (in average, it takes 271.87 s for the first model, and 1.80 s for the second one), and it decreased about 64% in number of constraints and 1.3% in number of variables, in average.

(n-v-T)	$cons_1$	$cons_2$	Gap (%)	var_1	var_2	Gap (%)	$CPU_1(s)$	$CPU_2(s)$
(2-3-15)	578	517	11.80	295	289	2.08	0.36	0.29
(3-3-25)	1030	858	19.95	556	547	1.65	1.23	0.49
(4-3-30)	1318	1029	28.03	757	745	1.61	2.00	0.82
(5-2-40)	1677	1370	22.39	913	903	1.11	3.08	1.07
(5-3-40)	1827	1370	33.36	1123	1108	1.35	4.46	1.07
(5-5-40)	2150	1370	56.89	1543	1518	1.65	4.12	1.62
(10-2-80)	4149	2735	51.69	2613	2593	0.77	375.67	2.58
(10-3-80)	4857	2735	77.57	3433	3403	0.88	188.57	2.56
(10-5-80)	6277	2735	129.48	5073	5023	1.00	390.72	3.22
(15-5-120)	12609	4100	207.54	10603	10528	0.71	1748.52	4.32
Average			**63.87**			**1.28**	**271.87**	**1.80**

Table 4.2 – Comparison between Model1 and Model2

4.4 Complexity analysis of sub problems

The complexity of a speed-scalable single machine scheduling problem, when the machine has just two states (ON and OFF), $(1, TOU|speeds, q_j|TEC)$ is already studied in [1]. The authors proved that the problem is NP-hard. So, the general version of this problem which is considered in this chapter, with three main states and two transition states, $(1, TOU|states, speeds, q_j|TEC)$ is also NP-hard.

In this section, we are interested to analyze the complexity of two sub versions of this problem. For this purpose, we considered the uniform speed case and the speed-scalable case of the problem when the sequences of the jobs are fixed.

4.4.1 $1, TOU|states, sequence, q_j|TEC$

problem statement

This section deals with a non-preemption scheduling problem of n jobs $(J_1, \cdots, J_n)$ on a multi-states single machine in a predefined order $(sequence = J_1 - J_2 - \cdots - J_{n-1} - J_n)$, with their related processing time (p_j) and energy consumption $(q_j)(j = 1, \cdots, n)$. This problem can be denoted by $1, TOU|states, sequence, q_j|TEC$, where '$TOU$' represents the time of use policy of energy price, '$states$' represents different states for the machine, '$sequence$' represents pre-defined sequence of the jobs, 'q_j' represents jobs' power demand, and 'TEC' represents the total energy cost.

To analyze the complexity of this problem, the same approach as what we did for the problems

$(1, TOU|states, sequence|TEC)$ and $(1, TOU|states, pmtn|TEC)$ in the previous chapters, is realized. So, in the following, a finite graph is proposed to model this problem.

Dynamic programming modelling approach

In this approach, the problem with T periods and n jobs is modelled by a graph with V nodes and E edges, which is composed of $T+1$ decision levels. Each level corresponds to a period of the time horizon (from 0 to T). A set of nodes (H_l) is allocated to each decision level (l) representing the possible last jobs in the given sequence (J_j), which has been processed until period l. The initial-OFF and final-OFF states are respectively specified by the nodes I and F. Each node is identified by a (k, l) notation, where, $k \in \{I, J_1, \cdots, J_n, F\}$ and $l \in \{0, \cdots, T\}$. Therefore, the graph is composed of $n+2$ different kinds of nodes. For example, in Fig. 4.5, $H_4 = \{I, J_1\}; H_{10} = \{J_1, J_2, J_3, F\}$.

Step 1: putting the nodes In this problem, the number of periods is fixed, and the required time for processing all the jobs, transitions and setups are given. So, the decision makers are facing x extra periods ($x = T - (P + \beta_1 + \beta_2 + 1)$, where $P = \sum_{j=1}^{n} p_j$), corresponding to x non-processing periods. That is why each node of the graph can appear in $x+1$ consecutive periods. Thus, sets τ_k are defined for any $k \in \{I, J_1, \cdots, J_n, F\}$, which contains the possible periods (decision levels) that machine can be in state k during them. For example, $\tau_I = \{0, \cdots, x\}$; $\tau_{J_1} = \{\beta_1 + p_1, \cdots, \beta_1 + p_1 + x\}$; $\tau_F = \{T - x, \cdots, T\}$. So, the total number of nodes for a problem with T periods and n jobs is:

$$|V| = (n+2) * (x+1) \cong nT \tag{4.19}$$

To illustrate the construction of this modelling approach, we consider an example with 3 with $n = 3, p_j = \{2, 1, 2\}, q_j = \{3, 5, 7\}, T = 15, \beta_1 = 2, \beta_2 = 1, x = 6$, and $c_t \in [0, 10]$. The first step of this approach for this example is presented in Fig. 4.3 which consists of 35 vertices.

Step 2: drawing the edges The edges of the graph can be divided into three sets (E_1, E_2, E_3). In this approach, the value of each edge ($Ev_{(k,l)-(k',l')}$), represents the total energy consumption cost (positive value) of related transition between two connecting nodes ((k, l) and (k', l')).

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

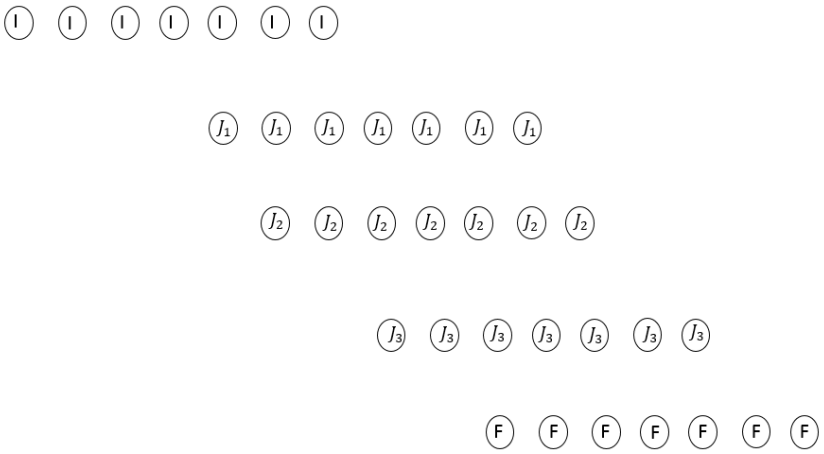

Figure 4.3 – Graph representation: step1

Step 2.1: E_1 The first set of edges (E_1) connects two nodes with the same k value in two consecutive decision levels (l and $l+1$). These edges can indicate the initial and final OFF state with the edge value of 0 (if $e_{OFF} = 0$) or the idle state with the edge value of:

$$Ev_{(k,l)-(k,l+1)} = c_{l+1} * e_{Idle} \quad ;\forall k \in \{J_1, \cdots, J_n\}; \forall l, l+1 \in \tau_k \tag{4.20}$$

where, c_l is the energy unit price in period l, and e_{Idle} is the machine's energy consumption in idle state. The cardinal of E_1 is $|E_1| = (n+1) \times x$.

Step 2.2: E_2 The second set of edges (E_2), connects a node k at period l with a node $k+1$ at period l' ($l' > l$), which illustrates three transitions cases:

- initial turning on and processing the first job (J_1) with the edge value of:

$$Ev_{(I,l)-(J_1,l')} = \sum_{i=l+1}^{l+\beta_1} (c_i * e_{Ton}) + \sum_{i=l+\beta_1+1}^{l+\beta_1+p_1} c_i * q_1 \quad ;\forall l \in \tau_I \tag{4.21}$$

- processing the job $k+1$ $(k = J_1, \cdots, J_{n-1})$:

$$Ev_{(k,l)-(k+1,l+p_{k+1})} = \sum_{i=l+1}^{l+p_{k+1}} c_i * q_{k+1} \quad ;\forall k \in \{J_1, J_2, \cdots, J_n\}; \forall l \in \tau_k \tag{4.22}$$

- final turning off:

$$Ev_{(J_n,l)-(F,l')} = \sum_{i=l+1}^{l'-1} (c_i * e_{Toff}) + c_{l'} * e_{OFF} \quad ;\forall l \in \tau_{J_n}; l' = l + \beta_2 + 1 \tag{4.23}$$

The cardinal of this set of edges is equal to $|E_2| = (n+1) * (x+1)$.

Step 2.3: E_3 The last set of the edges (E_3) corresponds to middle scheduling shutdowns between two processing jobs. Each middle shutdown consists of Toff, OFF, and Ton states. These edges connect nodes k in level l, and node $k+1$ in level l', where, $l' \in \{l+\beta_1+\beta_2+1+p_{k+1}, \cdots, l+x+1\} | l' \in \tau_{k+1}$ with the edge value of:

$$\begin{aligned} &Ev_{(k,l)-(k+1,l')} = \textstyle\sum_{i=l+1}^{l+\beta_2} (c_i * e_{Toff}) + \\ &\textstyle\sum_{i=l'-p_{k+1}}^{l'-p_{k+1}-\beta_1-1} (c_i * e_{Ton}) + \sum_{i=l'-p_{k+1}+1}^{l'} (c_i * q_{k+1}) \quad ;\forall k \in \{J_1, \cdots, J_{n-1}\}; \forall l \in \tau_k \end{aligned} \tag{4.24}$$

The total number of the third set of edges is equal to:

$$\begin{aligned} |E_3| &= \sum_{i=1}^{x-(\beta_1+\beta_2)} i * (n-1) \\ &= \frac{(x-(\beta_1+\beta_2)) * (x-(\beta_1+\beta_2)+1)}{2} * (n-1) \end{aligned} \tag{4.25}$$

Therefore, the total number of edges for a problem with T periods and n jobs is:

$$|E| = |E_1| + |E_2| + |E_3| \cong T^2 n \tag{4.26}$$

The second step of this approach for our considered example is presented in Fig. 4.4 which consists of 35 vertices and 64 edges.

Problem complexity analysis

Based on the presented modelling approach, each path of the graph which starts from node I in level 0 and ends at node F in level T, represents a feasible solution of the problem. Since the objective is to find the minimum total energy consumption costs, the shortest path that

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

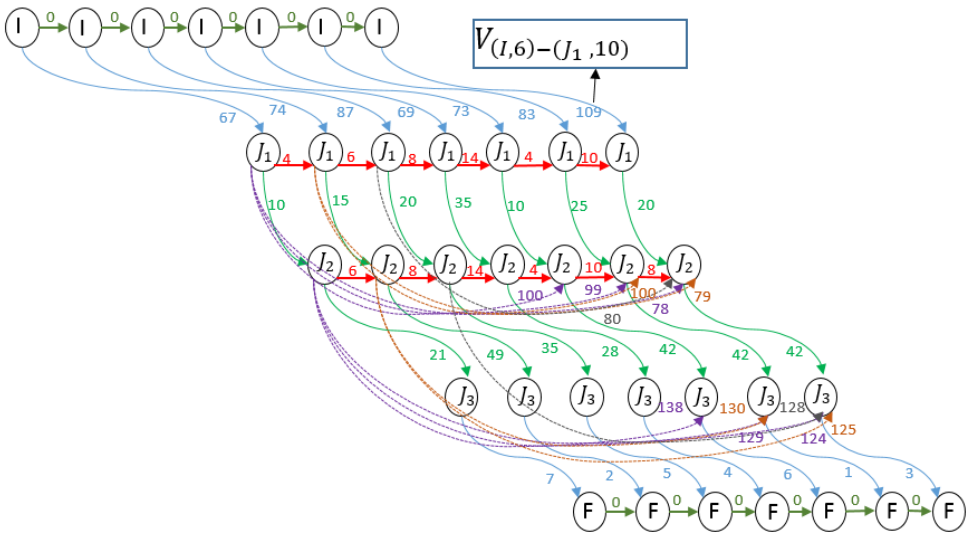

Figure 4.4 – Graph representation: step2

starts at node $(I, 0)$ and ends in node (F, T) represents the optimal solution of the problem. For this purpose, Dijkstra's algorithm is used to find the shortest path of the graph as the optimal solution.

Let us consider that the cost $C_{(k,l)}$ associated to node (k, l), indicates the minimum cost to obtain production level k at period l. The recurrence relationship used to evaluate it for each node, is as follows:

$$\begin{aligned} C_{(I,0)} &= 0 \\ C_{(k,l)} &= \min_{(k',l')\in A_{k,l}} \{C_{(k',l')} + V_{(k',l')-(k,l)}\} \end{aligned} \tag{4.27}$$

where $A_{k,l}$ is set of the precedent nodes connected to node (k, l) directly. For example, in Fig. 4.5 $A_{J_2,9} = \{(J_1, 4), (J_1, 8), (J_2, 8)\}$. Finally, $C_{(F,T)}$ represents the value of the optimal solution for the considered problem.

The application of Dijkstra's algorithm for the considered example, gives the optimal solution with turning on during period 1 and 2, then processing all the jobs based on their order, within period 3 to 7, and finally turning off at period 9, with the total cost of 133 (Fig. 4.5).

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

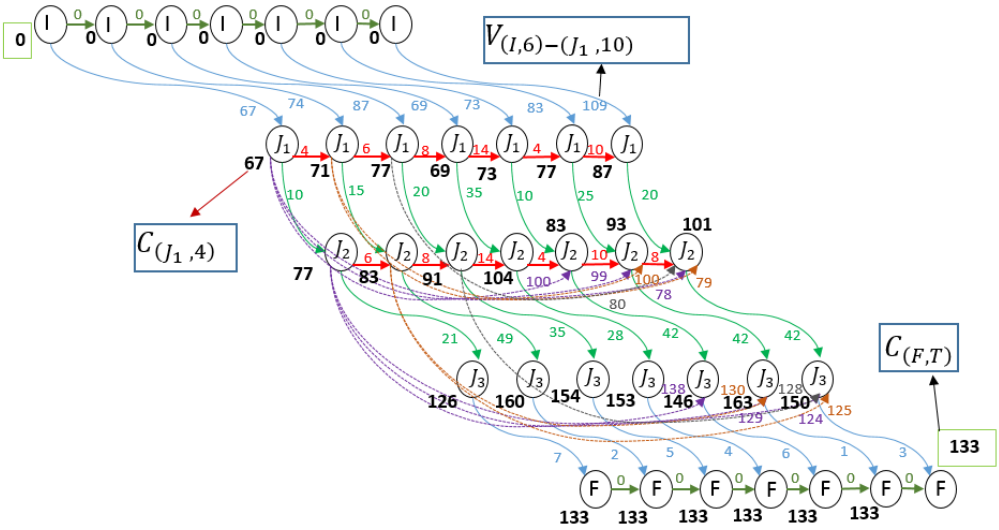

Figure 4.5 – Graph representation for an instance of uniform-speed problem

According to [77], the complexity of the developed approach is equal to:

$$\begin{aligned} &O(T^2n + Tn\log Tn) = O(T^2n + Tn\log T + Tn\log n) \\ &\cong O(T^2n) \end{aligned} \tag{4.28}$$

Since $n < T$ (worst case for any feasible problem), we have $\log n < \log T < T$. So, we can conclude that the problem is polynomial of degree 3 or a cubic polynomial problem ($O(T^3)$).

4.4.2 $1, TOU|states, speed, sequence, q_j|TEC$

problem statement

The second sub-problem, which is studied in this chapter, is a fixed sequence version of the general problem when the number of speeds is fix for all the jobs. Using the three-field notation, this problem can be denoted by:
$1, TOU|states, speed = v, sequence, q_j|TEC$, where '$speed = v$' represents the speed scalability of the machine when the same number of speeds are exist for all the jobs ($v_j = v_{j'} = v;\ \forall j, j'$). In this case, for each job, we have to choose its execution speed among a given set values. Each

speed corresponds to a given energy consumption and a processing time. Consequently, it could be interesting to process the jobs faster when the energy cost is low and process them slower when the energy cost is high.

Thus, assuming that the machine has v different processing speeds, each job j has v different possibilities for its processing time and power consumption. Without loss of generality, we assume that each job has a set of processing time and power consumption ($Q_j = \{(p_j^1, q_j^1), (p_j^2, q_j^2), \cdots, (p_j^v, q_j^v)\}$), with the following relations:

$$p_j^1 > p_j^2 > \cdots > p_j^v \quad ;\forall j \in \{1, \cdots, n\} \tag{4.29}$$

$$q_j^1 < q_j^2 < \cdots < q_j^v \quad ;\forall j \in \{1, \cdots, n\} \tag{4.30}$$

Note that to satisfy the non-preemption in this case, the solution must be composed of an unique speed i for each job j to process it with the related process time p_j^i and power consumption q_j^i non-preemptively. In the following, an adaptation of the proposed dynamic programming approach for uniform speed problem is used to model the speed scalable case.

Dynamic programming modelling approach

As the uniform-speed case of this problem, a graph with $T + 1$ decision levels is depicted. Each decision level (l) has a set of nodes (H_l) which represents the possible last jobs in the given sequence J_j, which is processed with speed i until moment l ($J_j^i \quad ;\forall j \in \{1, \cdots, n\}, i \in \{1, \cdots, v\}$) (Fig. 4.6). Therefore, by considering initial-OFF and final-OFF states, the graph is composed of $(n * v) + 2$ different kinds of nodes, where n represents the number of jobs, and v represents the number of speeds.

The number of non-processing periods for this case is obtained by the following formulation:

$$x' = T - \sum_{j=1}^{n} p_j^v - (\beta_1 + \beta_2 + 1) \tag{4.31}$$

In the whole graph, there are at most $(x' + 1)$ nodes with the same node's number ($k \in \{I, J_1^i, J_2^i, \cdots, J_n^i, F\}$). So, the total number of nodes for a problem with T periods, n jobs, and v speeds is:

$$|V| \leq ((n * v) + 2) * (x' + 1) \cong n \cdot v \cdot T \tag{4.32}$$

The first set of edges (E_1), indicates the initial and final OFF state with the edge value of 0 (if $e_{OFF} = 0$) or the idle state with the edge value of:

$$\begin{aligned} &Ev_{(k,l)-(k,l+1)} = c_{l+1} * e_{Idle} \\ &;\forall k \in J_j^i; \forall l, l+1 \in \tau_k; \forall j \in \{1, \cdots, n\}, i \in \{1, \cdots, v\} \end{aligned} \tag{4.33}$$

The cardinal of E_1 is:

$$|E_1| \leq [((n-1) * v) + 2] * x' \tag{4.34}$$

The second set of edges (E_2) illustrates three transitions cases:

- initial turning on and processing the first job with speed i (J_1^i) with the edge value of:

$$\begin{aligned} &Ev_{(I,l)-(J_1^i,l')} = \sum_{\theta=l+1}^{l+\beta_1} (c_\theta * e_{Ton}) + \sum_{\theta=l+\beta_1+1}^{l+\beta_1+p_1^i} c_\theta * q_1^i \\ &;\forall l \in \tau_I; \forall i \in \{1, \cdots, v\} \end{aligned} \tag{4.35}$$

- processing the jobs (except J_1):

$$\begin{aligned} &Ev_{(k,l)-(k+1,l+p_{k+1})} = \textstyle\sum_{\theta=l+1}^{l+p_{m+1}} c_\theta * q_{k+1} \\ &;\forall k \in \{J_1^i, J_2^i, \cdots, J_{n-1}^i\}; \forall l \in \tau_k; \forall i \in \{1, \cdots, v\} \end{aligned} \tag{4.36}$$

- final turning off:

$$\begin{aligned} &Ev_{(J_n^i,l)-(F,l')} = \sum_{\theta=t+1}^{l'-1} (c_\theta * e_{Toff}) + c_{l'} * e_{OFF} \\ &;\forall l \in \tau_{J_n^i}; \forall i \in \{1, \cdots, v\}; l' = l + \beta_2 + 1 \end{aligned} \tag{4.37}$$

The cardinal of this set of edges is equal to:

$$|E_2| \leq [((n-1) * v) + 2] * x' \tag{4.38}$$

The last set of the edges (E_3) corresponds to middle scheduling shutdowns between two processing jobs, and processing the second ones. These edges connect nodes k in level l, and node $k+1$ in level l', where, $l' \in \{l + \beta_1 + \beta_2 + 1 + p_{k+1}, \cdots, l + x + 1\} | l' \in \tau_{k+1}$ with the edge value of:

$$\begin{aligned} &Ev_{(k,l)-(k+1,l')} = \textstyle\sum_{\theta=l+1}^{l+\beta_2} (c_\theta * e_{Toff}) + \sum_{\theta=l'-p_{k+1}-\beta_1-1}^{l'-p_{k+1}} (c_\theta * e_{Ton}) \\ &+ \textstyle\sum_{\theta=l'-p_{k+1}+1}^{t'} (c_\theta * q_{k+1}) \quad ;\forall k \in \{J_1^i, \cdots, J_{n-1}^i\}; \forall l \in \tau_k; \forall i \in \{1, \cdots, v\} \end{aligned} \tag{4.39}$$

The total number of these edges is equal to:

$$\begin{aligned} |E_3| &= \sum_{\theta=1}^{x-(\beta_1+\beta_2)} \theta * (n-1) * v \\ &= \frac{(x-(\beta_1+\beta_2)) * (x-(\beta_1+\beta_2)+1)}{2} * (n-1) * v \end{aligned} \tag{4.40}$$

Therefore, the total number of edges for a speed scalable problem with T periods, n jobs, and v speeds is:

$$|E| = |E_1| + |E_2| + |E_3| \cong T^2 n v^2 \tag{4.41}$$

A part of this graph for a problem with T periods, n jobs, and v possible processing speeds for each job is illustrated in Fig. 4.6.

Problem complexity analysis

As it is mentioned in previous section, the shortest path that starts at node $(I, 0)$ and ends in node (F, T) represents the optimal solution for the speed scalable problem too. In this case, the recurrence relationship for evaluating the cost of each node is the same as uniform-speed case (equation 9).
According to [77], the complexity of Dijkstra's algorithm for the speed-scalable problem can be calculate by the following formulation:

$$O(T^2 nv + Tnv \log Tnv) \tag{4.42}$$

Since $n < T$, we have $\log n < \log T < T$, also $\log v < v$, and there is not any limitation for v. So, we can conclude that the speed-scalable case of this problem is at most pseudo polynomial.

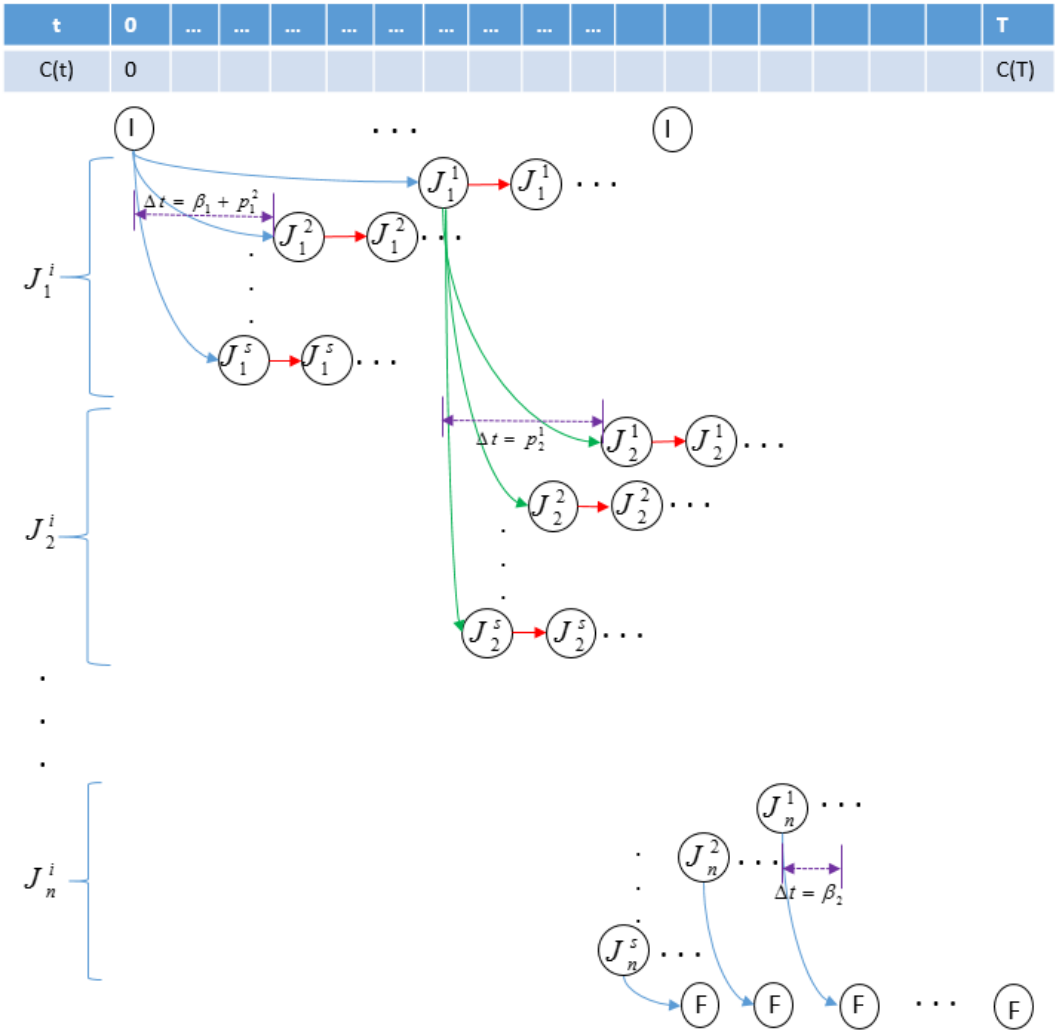

Figure 4.6 – Graph representation for a speed scalable problem.

4.5 Optimization methods

As it is presented in the previous sections of this chapter, the scheduling problem of some jobs on a multi-states and multi-speeds single machine when the jobs consumed different amount of energy is NP-hard.

The exact methods are not able to solve the large size instances of the NP-hard problems during a reasonable time. For this reason, we proposed a heuristic algorithm as well as a genetic algorithm to solve this problem. The description of these two algorithms and their performance comparison are presented in the following.

4.5.1 Heuristic method

In this study, it is attempted to present an intelligent and effective heuristic algorithm for the considered problem to obtain a feasible solution of the large size instances in a reasonable time. This heuristic's principles are based on the allocating the non-processing states to the machine during the extra periods (the same idea as the presented heuristic in the previous chapter (section 3.7.1)).

Unlike the considered problem in the previous chapter, in this chapter different possible processing times are assumed for each job. Therefore, to obtain the maximum number of extra periods which computes by equation 5.1, the minimum required number of periods for processing times must be replaced to P. For this purpose, the processing time of each job with the maximum speed (p_{j,v_j}) is selected. Thus, by considering the assumed parameters, the value of x can be obtain as:

$$x = T - (\beta_1 + \beta_2 + 1) - \sum_{j=1}^{n} p_{j,v_j} \tag{4.43}$$

For example, in a problem with 3 jobs, 30 periods, and 3 speeds, with the parameters values as follow: $p_1 = \{6, 4, 2\}, p_2 = \{5, 4, 3\}, p_3 = \{5, 3, 1\}, \beta_1 = 2, \beta_2 = 1$, the x value is equal to [30-(2+1+1)-(2+3+1)]=20.

Heuristic's procedure

The presented algorithm is composed of two general steps (forward and backward steps) which specify the machine's state period by period to consider all the possibilities. In each step, the algorithm investigates $(x+1)$ different situations, and selects the solution with minimum objective function. Thus, to obtain the best solution, $2*(x+1)$ different solutions must be considered and compared with each other. Finally, the solution with the minimum objective value between the best solutions of the forward and the backward steps will be selected as the heuristic's solution. Since, number of solutions to be checked is dependent on x value, so, this method reduces dependency of the algorithm to number of jobs, periods and speeds (size of the problem) and increases its efficiency.

Forward step For each solution, in the forward step, a specific amount $(i = 0, ..., x)$ of the extra periods is allocated to the initial-OFF states at the first stage. Then, the machine must be in Ton state during periods $i+1$ to $i+\beta_1$. For example, for the presented instance in figure 4.7, in the fourth solution OFF state is considered for the machine from period 0 to 3. When the machine is in ON state, it must process one job, so, the algorithm selects the job with the minimum average energy consumption costs per period. As can be seen in figure 4.7, in the fourth solution, the machine is in ON state from period 6 and $j1$ is selected to process with speed 1. After completing the first job, while all the extra periods are not attributed, the algorithm selects the state of the machine with the minimum average energy consumption cost per period among ON, middle-OFF, and Idle states. Note that, any middle-OFF state consists of Toff, OFF, and Ton states, which needs at least $\beta_2 + \beta_1 + 1$ periods. These steps are repeated until all the jobs are performed and the machine's state is specified during all the periods. Once the algorithm's processes is finished, the objective value of the solution is computed (see Table 4.3, steps 1.1 to 1.6).

Backward step For the backward step, the same procedures as forward step are performed, but by allocating a specific amount of extra periods $(i = 0, ..., x)$ to the final-OFF states during the final periods. For example, for the presented instance in figure 4.8, in the fourth solution OFF state is considered for the machine from period 12 to 15. Then, the machine must be in Toff state

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	TEC
c_t	0	2	3	5	4	3	6	7	5	5	2	4	3	6	9	6	
1	**OFF**	**Ton**	**Ton**	J1-1	J1-1	J1-1	J1-1	Toff	OFF	OFF	Ton	Ton	J2-3	J2-3	Toff	**OFF**	161
2	**OFF**	**OFF**	**Ton**	**Ton**	J1-1	J1-1	J1-1	J1-1	Toff	OFF	Ton	Ton	J2-3	J2-3	Toff	**OFF**	178
3	**OFF**	**OFF**	**OFF**	**Ton**	**Ton**	J1-1	J1-1	J1-1	J1-1	Idle	Idle	Idle	J2-3	J2-3	Toff	**OFF**	172
4	**OFF**	**OFF**	**OFF**	**OFF**	**Ton**	**Ton**	J1-1	J1-1	J1-1	J1-1	Idle	Idle	J2-3	J2-3	Toff	**OFF**	156
5	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**Ton**	**Ton**	J1-1	J1-1	J1-1	J1-1	Idle	J2-3	J2-3	Toff	**OFF**	154
6	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**Ton**	**Ton**	J1-1	J1-1	J1-1	J1-1	J2-3	J2-3	Toff	**OFF**	160
7	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**Ton**	**Ton**	J1-2	J1-2	Idle	J2-3	J2-3	Toff	**OFF**	152
8	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**Ton**	**Ton**	J1-3	Idle	J2-3	J2-3	Toff	**OFF**	129
9	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**Ton**	**Ton**	J1-3	J2-3	J2-3	Toff	**OFF**	114

Figure 4.7 – forward step

during periods $T-i-\beta_2$ to $T-i-1$, and in ON states from period $T-i-\beta_2-1$. For example, in the $(x+1)th$ solution of the backward step, the machine is allocated in the final OFF states during periods $t=T-x,...,T$, and it is in the Toff state during periods $t=T-x-\beta_2,...,T-x-1$ (see Table 4.3, steps 2.1 to 2.4).

The procedures of this algorithm is represented in Table 4.3.

Illustrative example

For better understanding these procedures, the algorithm is applied to a small example with 2 jobs, 15 periods, and 3 speeds. The processing times and energy consumptions of each job, and the other parameters' values of this example are presented in Table 4.4. According to these values, $x=8$. So, 9 different solutions must be compared by the algorithm in each step (18 solutions in total). These solutions and their objective value (total energy cost) are presented in figures 4.7 and 4.8. Where, $Jj-i$ corresponds to process job j with speed i. The bold texts are for the states of the machine which are fixed by algorithm for each solution, and the normal texts are the states which are chosen by algorithm procedures. As it can be seen, the best solution of the forward step is the last one with the objective value of 114, while, among the backward's solutions the best solution has the objective value of 97. Therefore, the best scheduling plan found by the proposed heuristic algorithm for this problem is the last solution of backward step with total energy cost equal to 97.

Heuristic Algorithm : HA

if $(x < 0)$ **then** The problem is infeasible.
else if $(x = 0)$ **then**
All the jobs must be performed continuously with the maximum speed
in any processing order (with the same objective values).
else
Step 1: for $(i = 0$ to $x)$
1.1. Put the machine in initial OFF states for periods $t \in [0, i] \cup T$.
1.2. Turn the machine on just after period i.
1.3. Process the job which needs the minimum average of
energy consumption costs per period.
1.4. Choose the machine's state with the minimum average of energy consumption costs per period among {ON, Idle (if $x - i > 0$), middle OFF (if $x - i > \beta_2 + \beta_1$)}.
if ON state is selected **then** go back to the stage 1.3.
else if Idle state is selected **then**
put the machine in Idle; $x - i = x - i - 1$; return to the stage 1.4.
else middle OFF state is chosen **then**
Turn the machine off for the $(x - i)$ next periods, (consider Toff,
OFF and Ton steps); $x - i = 0$; return to the stage 1.4.
end if
1.5. **if** $(x - i = 0)$ **then**
Perform the remaining jobs based on the numerical order with the maximum speed.
end if
1.6. Compute the objective function for each solution (Z_i)
end for
let Z_{min_1}=$min\{Z_i\}$
Step 2: for $k = 0$ to x
2.1. Put the machine in final OFF state for periods $t \in [T - k, T] \cup 0$.
2.2. Turn on the machine for the periods before $(t = T - k - 1$ to $t = T - k - \beta_2)$.
2.3. Do the same as step 1.3 to 1.5 in the backward ways.
2.4. Compute the objective function for each solution (Z_k).
end for
let Z_{min_2}=$min\{Z_k\}$
let Z=$min\{Z_{min_1}, Z_{min_2}\}$ (minimum objective value of all the solutions)
end if

Table 4.3 – The heuristic algorithm

Parameter	Value	Parameter	Value	Parameter	Value
$p_{1,1}$	4	$p_{1,2}$	2	$p_{1,3}$	1
$p_{2,1}$	5	$p_{2,2}$	3	$p_{2,3}$	2
$q_{1,1}$	2	$q_{1,2}$	3	$q_{1,3}$	4
$q_{2,1}$	3	$q_{2,2}$	4	$q_{2,3}$	6
β_1	2	β_2	1	e_{OFF}	0
e_{Idle}	2	e_{Ton}	5	e_{Toff}	1

Table 4.4 – The parameters' values for an instance with 2 jobs, 15 periods and 3 speeds

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	TEC
c_t	0	2	3	5	4	3	6	7	5	5	2	4	3	6	9	6	
1	**OFF**	Ton	Ton	J2-3	J2-3	Idle	Idle	Idle	Idle	Idle	J1-1	J1-1	J1-1	J1-1	**Toff**	**OFF**	170
2	**OFF**	Ton	Ton	J2-3	J2-3	Idle	Idle	Idle	Idle	J1-1	J1-1	J1-1	J1-1	**Toff**	**OFF**	**OFF**	155
3	**OFF**	Ton	Ton	J2-3	J2-3	Idle	Idle	Idle	J1-1	J1-1	J1-1	J1-1	**Toff**	**OFF**	**OFF**	**OFF**	146
4	**OFF**	Ton	Ton	J2-3	J2-3	Idle	Idle	Idle	Idle	Idle	J1-3	**Toff**	**OFF**	**OFF**	**OFF**	**OFF**	143
5	**OFF**	Ton	Ton	J2-3	J2-3	Idle	J1-1	J1-1	J1-1	J1-1	**Toff**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	133
6	**OFF**	Ton	Ton	J2-3	J2-3	J1-1	J1-1	J1-1	J1-1	**Toff**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	126
7	**OFF**	Ton	Ton	J2-3	J2-3	Idle	J1-2	J1-2	**Toff**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	129
8	**OFF**	Ton	Ton	J2-3	J2-3	J1-2	J1-2	**Toff**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	113
9	**OFF**	Ton	Ton	J2-3	J2-3	J1-3	**Toff**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	**OFF**	97

Figure 4.8 – backward step

4.5.2 Genetic Algorithm

As we used a genetic algorithm in the previous chapter to evaluate the performance of our heuristics for the problem with the same energy consumption of the jobs, we also developed a genetic algorithm for the studied problem in this chapter. Genetic algorithms are a population-based Meta heuristics. The implementation of a genetic algorithm (figure 3.16) needs to establish several concepts such as chromosome, initial population, parent selection method, crossover and mutation operators. These concepts for our problem are defined in the following sections.

Initial population and Chromosome representation

Any solution of this problem, is a schedule over a time horizon from period 0 to period T, that defines the machine's state during each period. So, in this paper, each chromosome of the genetic algorithm is represented by $T + 1$ genes and each gene identifies the machine's state in a period. To distinguish the machine's states, each state is represented by a specific number as $OFF = 1$, $Ton = 2$, $Idle = 3$, and $Toff = 4$. Besides, an integer number greater than 100

1	1	1	1	2	2	110	110	110	230	230	320	320	320	320	4	1	1	1	2	2	430	430	520	520	520	4	1	1	1	1	1	1

Figure 4.9 – The chromosome of our genetic algorithm

($k > 100$) represents that the machine is in ON state. In the other words, if in period t, the machine processes the job j with speed i, in the related chromosome, the gene t fills with number $(100 * j + 10 * i)$. Figure 5.8 represents the corresponding chromosome of the presented instance at Figure 4.2. Since in this instance the number of periods is 32, so this chromosome consists of 33 genes. The number 230 in 10th gene means that during period 9 the machine processes job 2 with speed 3. Also, the number 4 in 27th gene means that during period 26 the machine is in Toff state.

The genetic algorithm's procedure starts with a randomly generated initial population. In this study, the proposed genetic algorithm uses an initial population which is randomly generated based on the same idea as our heuristic. In other words, for each individual, at first, the period that the machine is in the first Ton state must be selected randomly. Then, the job's number and its processing speed will be chosen randomly. After completing the job, the machine's state must be selected among ON, Toff and Idle states arbitrarily. These procedures must be continued to process all of the jobs until the last period. Finally, the objective value of the problem will be computed as a fitness function to classify the quality of the generated chromosome.

For completing the genetic algorithm's procedure, it is necessary to choose it's main parameters based on the studied problem. These parameters are the population size, the crossover rate, the mutation rate, a crossover operator, a mutation operator and the number of iterations. The performance of the algorithm depends on the selection of these parameters.

Crossover and mutation

In this study, the roulette wheel selection operator has been chosen as parents selection operator to produce the new offsprings. In the literature, two approaches are mostly used as the crossover operator such as single-point and double-point. For this purpose, the cut point(s) will be randomly generated from the period's index.

Single-point method In the single-point method, the first offspring will be composed of the first parent from the beginning to the cut point, and the second parent from the first gene after the cut point to end of the chromosome. The second offspring will be obtained by the same way and reverse the parents.

After producing the children, a correction procedure must be done to convert the not feasible solutions to the feasible one. For example, as it is shown in Figure 4.10, the first children has two processing speed for job 3. There is a preemption for job 3, and job 4 is not performed. So, the solution is not feasible and it requires some corrections. For this purpose, we start from the first gene and we correct the chromosome when it is necessary. For the presented child in Figure 4.10 (ch1), in 12th gene the machine must perform job 3 with speed 2 which takes 4 periods, so, from 12th to 15th gene must be fill with 320. Then, the next job which is chosen is job 5 with speed 2 that can be performed in 3 periods (from 16th to 18th gene). Since job 4 is not processed yet, we will perform it with the maximum speed (3). Therefore, gene 19 and 20 must fill with 430. Finally, once all the job are finished, we can put the machine in Toff and OFF states for the rest.

Double-point method In the double-point method, the first offspring will be composed of the first parent from the beginning to the first cut point, the second parent from the next gene after the first cut point to the second cut point, and the first parent from the next gene after the second cut point to the end of the chromosome. The second offspring will be obtained by the same way and reverse the parents. Finally, a correction procedure is also required to convert the not feasible solutions to the feasible one with respecting to the obtained states order and processing order (Figure 4.11).

For the presented example in Figure 4.11, the first obtained children (ch1) is correct until 9th gene. Therefore it will be the same in the corrected children (child1). Since the required processing time for performing the second job with the third speed (230) is equal to 2, so the 10th gene of child1 must be fill by 230. Just after the 9th gene in (ch1), 10th gene is 1, which means the machine is putted in OFF state, but in the studied system for turning the machine off, one Toff state (4) is required. Therefore, 11th gene of child1 must be fill by 4. Then, from 12th gene to 19th gene of child1 are the same as 10th gene to 17th gene of ch1. Since the machine is able to process the third job with the first speed (310) in 5 periods, so, 20th gene must be fill with (310).

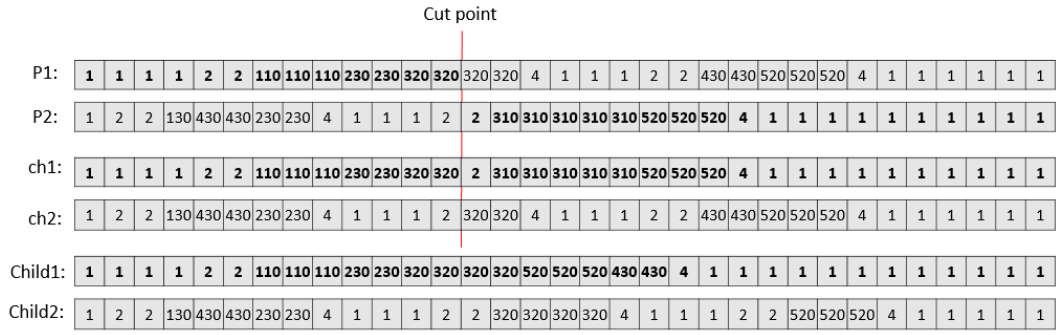

Figure 4.10 – single-point crossover method

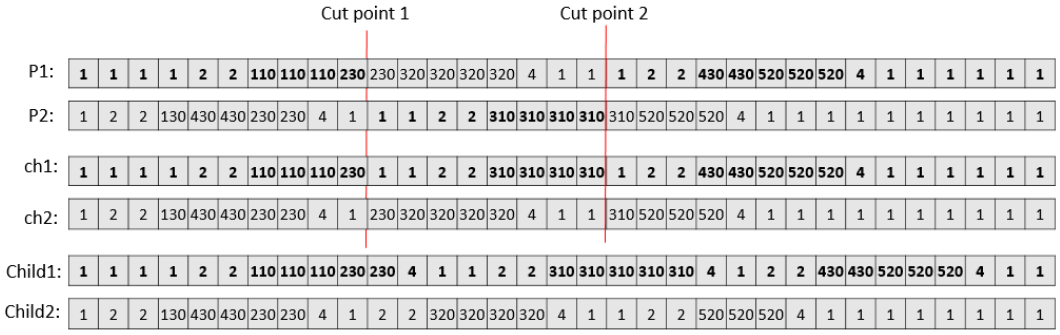

Figure 4.11 – Double-point crossover method

For turning off the machine, 4 is putted in 21th gene of child1. From 22th gene to the end are the same as 18th-28th genes of ch1.

Mutation method For the mutation method, a chromosome from the initial population will be randomly selected and the mutation will be performed on the selected gene by swapping its value. Then, it is necessary to check the feasibility of the obtained offsprings and correct the not feasible ones like the crossover method.

The final step of each iteration is to update the initial population for the next iteration. For this purpose, the best chromosomes in terms of the fitness function must be selected from all the initial population and the obtained chromosomes by crossover and mutation methods in the previous iteration.

In this thesis, a Taguchi orthogonal array is utilized instead of a full factorial experimental

Factor	level1	Level2	Level3
Crossover operator	single-point	double-point	-
Mutation operator	swap	revers	insert
Crossover rate	0.7	0.8	0.9
Mutation rate	0.05	0.10	0.15
Population size	50	100	150
Iteration number	50	100	150

Table 4.5 – Design factors and their levels

design for determining the parameters of the genetic algorithm. This approach is presented in the following.

Taguchi method for parameters setting

An experimental design method is developed by Genichi Taguchi to increase the efficiency of implementation and evaluation of experiments. In Taguchi method, experimental results are converted to a signal/noise (S/N) ratio, which can be calculated in three different ways, such as 'small value is good', 'great value is good' and 'nominal value is good'. By this way, a level of the factor which has the highest ratio represents a better performance.

Design factors and their levels

In this study, Taguchi method is applied to reduce the number of experiments. For this purpose, six factors are considered and one of them has two levels and the others have three levels (Table 5.1). So, we applied the L_{18} (one two-level and up to seven three-levels) orthogonal array. Moreover, to achieve accuracy, experiments were repeated five times for each problem.

Data analysis

The results of taguchi method analysis are given in Figure 5.9. As a result, single-point is selected as crossover method to produce two new children, and the crossover rate is selected equal to 0.7 (70%). The mutation rate is selected equal to 0.1 (10%), and the initial population size and the number of iterations are considered equal to 150 and 100, respectively.

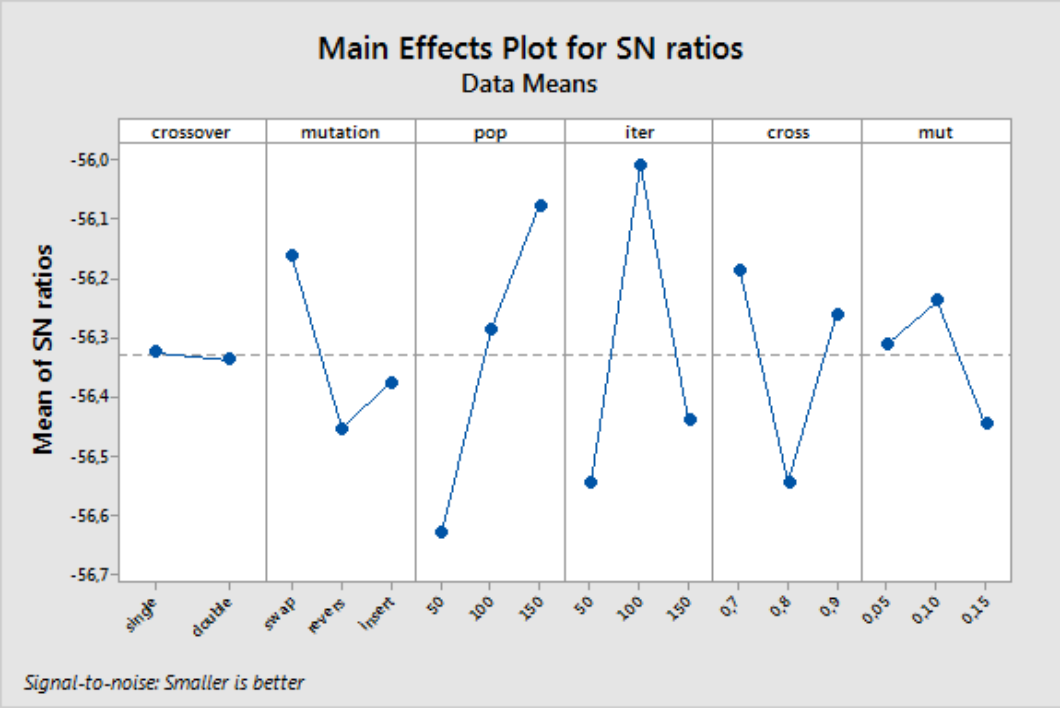

Figure 4.12 – Taguchi analysis result

4.5.3 Memetic Algorithm

Premature convergence is an inherent characteristic of the classical genetic algorithms that makes them incapable of searching numerous solutions of the problem domain. A memetic algorithm is an extension of the traditional genetic algorithm. It uses a local search technique to reduce the likelihood of the premature convergence ([81]). Therefore, to improve the quality of the obtained solutions by the genetic algorithm for this problem, a local search procedure is also introduced (memetic algorithm). The local search procedure is applied to increase the quality of the 15 (10% of the population size) best chromosomes of the population in each iteration. For this purpose, a new solution will be created by increasing the processing speed one unit for a job, and consequently, performing all the remaining jobs earlier as much as the difference between the related processing times. This procedure must be repeated for all the jobs in their sequence order. So, for a problem with n jobs, at most n new solutions can be created from each initial solution. Finally, the solution with the best objective function must replace the initial one. For example, for the considered chromosome in Figure 4.13, the first job was processed with the first

1	1	1	1	2	2	110	110	110	230	230	320	320	320	320	4	1	1	1	2	2	420	420	420	520	520	520	4	1	1	1	1	1
1	1	1	1	2	2	120	120	230	230	320	320	320	320	4	1	1	1	2	2	420	420	420	520	520	520	4	1	1	1	1	1	1

Figure 4.13 – The local search procedure for an example

At the end of each iteration find the 15 best solutions;
For ($i = 1$ to 15)
  Get the initial solution x_i;
  Find the sequence order of the job in the solution x_i;
  For ($j = 1$ to n)
    If ($speed_j < speed_{max}$) **Then**
      Create the new solution (y_i), by doing the job j with speed $speed_j + 1$,
      then do the remaining jobs respectively and turn off the machine after the last job;
    End If
    compute the objective function for y_i;
    If ($objective(y_i) < objective(x_i)$) **Then**
      Replace the first solution (x_i) by the new one (y_i);
    End If
  End For
End For

Table 4.6 – Local search's procedure

speed. In the first proposed solution by the memetic algorithm, the first job is processed with the second speed, and then, all the jobs are done earlier. The pseudo code for this local search is given in Table 4.6.

4.5.4 Numerical experiments

The performances of the proposed methods in this chapter have been examined by several numerical instances inspired by the literature ([61]). For this purpose, the heuristic algorithm, the genetic algorithm and the memetic algorithm have been coded by C++ language in the Visual Studio 2015, and CPLEX software is used to solve the instances with the exact method (Branch and Cut). Five different examples are randomly generated for each instance size by changing the processing times and the energy consumptions of the jobs, as well as the unit of energy price in each period. The computation time with CPLEX, for all the experiments was set to 1 hour or 3600 seconds. For the problems smaller than 15 jobs, 5 speeds, and 120 periods, CPLEX was able to find the optimal solutions. Therefore, the results of the proposed algorithms are compared with the optimal solution. These results are presented in Table **??**.

The gap between the objective function of each algorithm and the exact method as well as their computation time are presented as the results. Moreover, the numerical results have been illustrated by minimum, average, and maximum obtained value of each criterion for each problem size. In general, for the small size problems, the heuristics find the solutions with the gap of 17.1% in average, while this gap for the genetic algorithm and the memetic algorithm is equal to 7.5% and 2.7%, respectively. The average computation time of small size problems for heuristic, genetic, and memetic algorithms are equals to 1.06 s, 15.64 s and 18.50 s, respectively.

For the problems larger than 20 jobs, 5 speeds and 160 periods, CPLEX was not able to find the optimal solution. So, in Table **??**, we just compared the obtained solutions by these three algorithms. It must be mention that, in all the cases, the memetic algorithm finds the best-obtained solution. As it is presented in Table **??**, the average gap between heuristic and genetic algorithms' solutions and the obtained solution by MA, is about 21.1% for HA, and 18.4% for the GA. The average computation time of these problems for heuristic, genetic, and memetic algorithms are equals to 7.82 s, 34.37 s, and 61.59 s, respectively.
Moreover, an analysis of the variance (ANOVA) with a confidence level of 95% was taken using the Minitab.17 software to check the statistical validity of the results (Figures 4.14 and 4.15).

(n,v,T)		$Gap_{HA-Cplex}$	$Gap_{GA-Cplex}$	$Gap_{MA-Cplex}$
(2,3,15)	Min	0.0	0.0	0.0
	Average	**1.6**	**0.0**	**0.4**
	Max	4.3	0.0	1.1
(3,3,25)	Min	0.6	0.6	0.6
	Average	**13.9**	**3.0**	**1.5**
	Max	26.1	4.3	2.2
(4,3,30)	Min	5.9	2.1	0.0
	Average	**14.6**	**4.1**	**0.6**
	Max	25.5	9.7	2.1
(5,2,40)	Min	10.0	0.0	0.0
	Average	**17.3**	**1.6**	**1.1**
	Max	31.4	3.1	1.7
(5,3,40)	Min	7.9	0.0	0.0
	Average	**10.3**	**2.5**	**1.0**
	Max	16.4	5.2	3.1
(5,5,40)	Min	8.4	0.2	0.2
	Average	**22.7**	**2.5**	**0.8**
	Max	36.9	5.1	1.5
(10,2,80)	Min	0.1	5.3	0.3
	Average	**12.6**	**9.8**	**1.1**
	Max	21.9	17.7	2.0
(10,3,80)	Min	5.0	7.6	1.6
	Average	**21.1**	**14.1**	**3.7**
	Max	33.4	20.1	6.2
(10,5,80)	Min	26.8	13.9	2.8
	Average	**33.6**	**21.1**	**7.7**
	Max	40.6	28.6	11.0
(15,5,120)	Min	21.3	12.5	4.5
	Average	**23.6**	**15.9**	**8.7**
	Max	25.4	18.2	11.60
Average Gap		**17.1**	**7.5**	**2.7**
Average CPU (s)	**Cplex=271.01**	**1.06**	**15.64**	**18.50**

Table 4.7 – Comparison between the obtained solutions by the proposed methods and the CPLEX

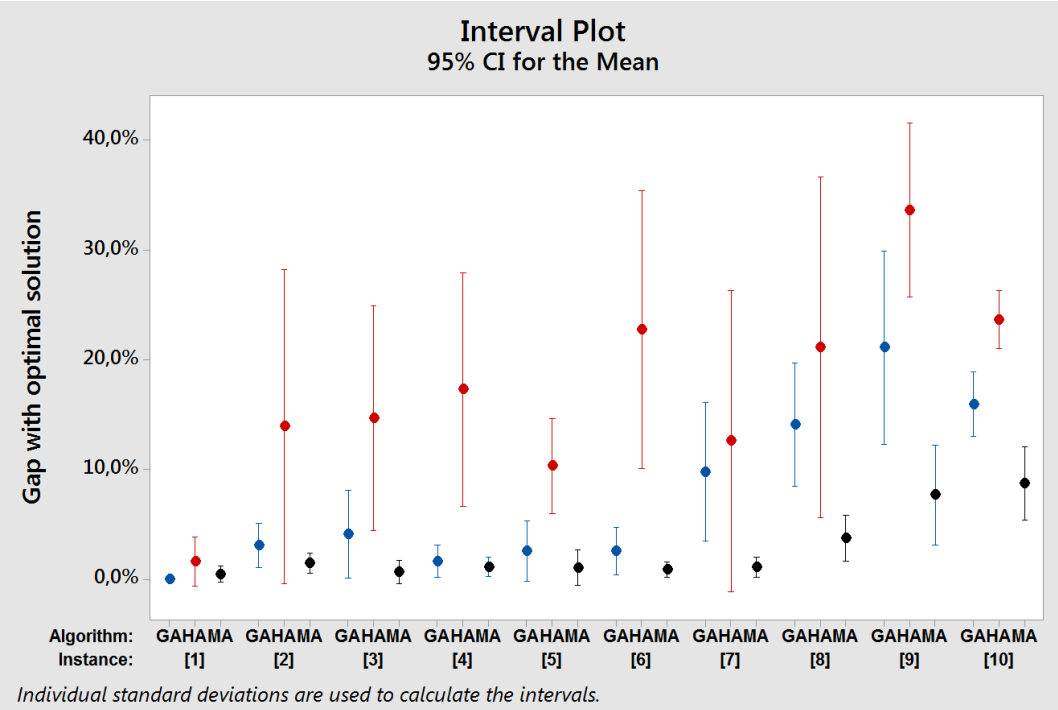

Figure 4.14 – Performance comparison of GA, HA, and MA with the optimal solution

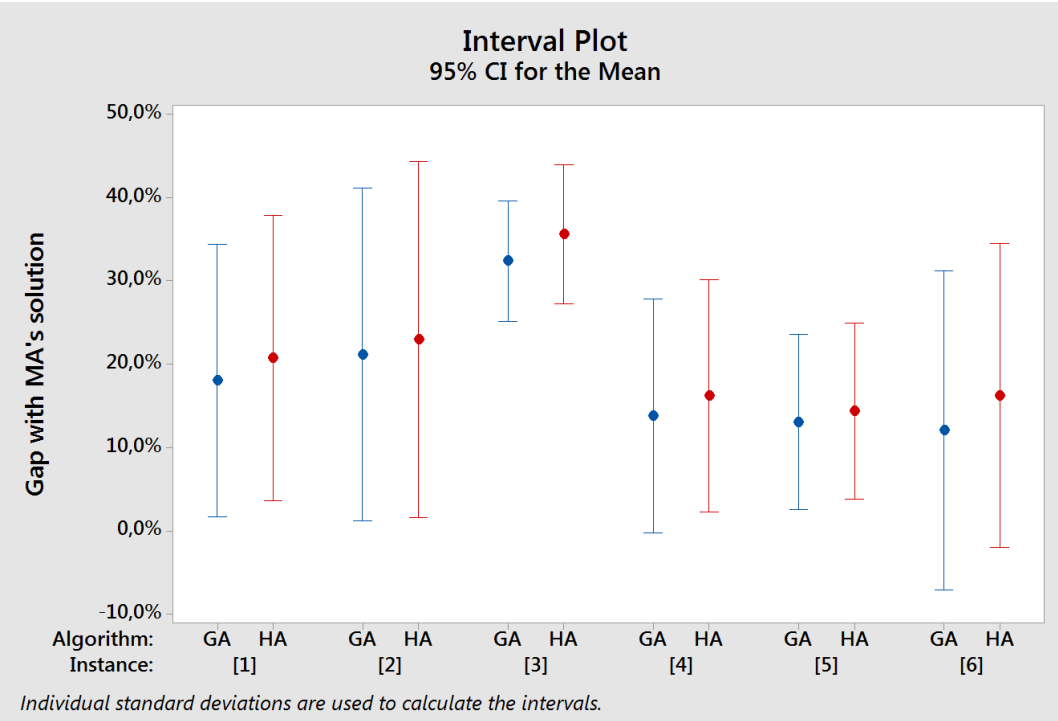

Figure 4.15 – The results improvement by MA against GA and HA for the large size instances

(n,v,T)		$Gap_{HA-MA}(\%)$	$Gap_{GA-MA}(\%)$
(20,5,160)	Min	6.2	3.0
	Average	**20.7**	**18.0**
	Max	37.4	33.3
(25,5,200)	Min	4.7	3.6
	Average	**22.9**	**21.1**
	Max	39.6	37.8
(30,5,240)	Min	24.7	23.8
	Average	**35.6**	**32.4**
	Max	41.1	37.5
(35,5,280)	Min	8.6	6.1
	Average	**16.2**	**13.8**
	Max	34.4	33.0
(40,5,320)	Min	4.5	4.5
	Average	**14.4**	**13.0**
	Max	26.1	25.2
(45,5,360)	Min	4.0	1.8
	Average	**16.2**	**12.1**
	Max	40.7	39.0
Average Gap		**21.1**	**18.4**
Average CPU (s)	**MA=61.59 s**	**HA=7.82 s**	**GA=34.37 s**

Table 4.8 – The results improvement by MA comparing to GA and HA

4.6 Conclusion

A scheduling problem of several jobs with different energy consumptions on a speed scalable and multi-states single machine scheduling problem is addressed in this chapter to minimize the total energy consumption costs. Our first contribution consists of proposing two new mathematical formulations to model the problem. Then, since the considered problem is known to be NP-hard, a heuristic, a genetic and a memetic algorithms are presented to solve the medium and large size problems in a reasonable time. Moreover, the complexity of two sub-problems when the jobs must be performed in a predetermined order, are studied. For this purpose, the dynamic programming approaches are used to solve the problem. As the results, we demonstrated that the problem with uniform speed in polynomial, and the speed-scalable problem is pseudo-polynomial. The next chapter will present a summary of all the conclusions of this thesis.

5

Conclusion and perspectives

This thesis focuses on modeling and optimization of three types of multi-states single machine scheduling problems arising in industry to minimize their total energy consumption costs under time varied energy prices. For this purpose, firstly a complete study on the different scheduling problems with the energy concepts is performed. The studied problems are divided into four main sets and the brief descriptions of these papers are presented in chapter 1.

As the first contribution of this thesis, we proposed an improved mathematical model for an existing problem in the literature ([61]), and we analyzed the complexity of the considered problem by using a new dynamic programming approach in chapter 2. Regarding to the results, we proved that unlike what is considered before in [61], the scheduling problem of several jobs with a predetermined sequence on a multi-states single machine $(1, TOU|states,\ sequence|TEC)$ is polynomial.

In chapter 3, we studied a general version of the previous problem to optimize the sequence order of the jobs, as well as, the machine's state in each period $(1, TOU|states|TEC$ problem). We presented a mathematical model for this problem and we proved it's NP-hardness using a 3-PARTITION problem. Then, we proposed a heuristic and a genetic algorithms able to find the near optimal solutions for any size of problems in few minutes. We also defined four lower bounds for this problem. Moreover, the complexity of four sub-problems is also investigated. As the results, we proved that all these problems $(1, c_t = c|states|TEC$, $1, c_t < c_{t+1}|states|TEC$, $1, TOU|states,\ pmtn|TEC$ and $1, TOU|states,\ p_j = p|TEC)$ are polynomial problems.

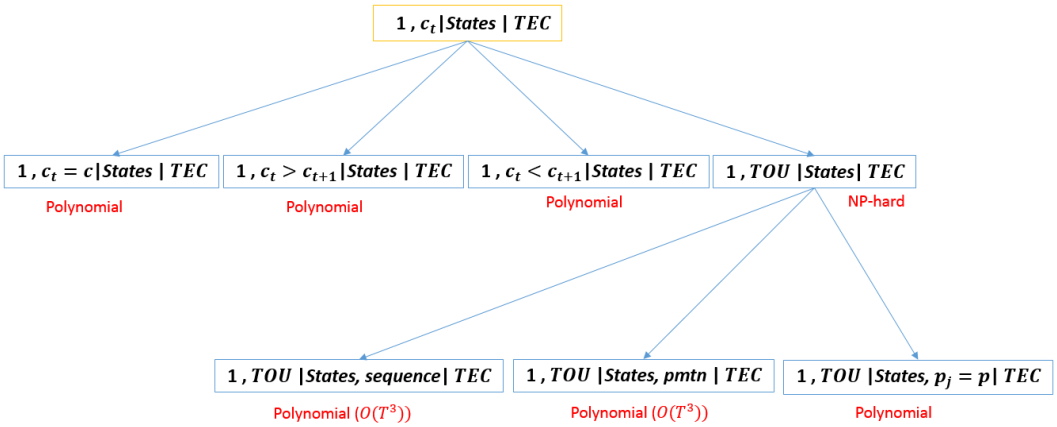

Figure 5.1 – Our contributions for different variants of $1, c_t|states|TEC$ problem

In chapter 4, we studied a more general scheduling problem with different energy consumption for the jobs and different speed for the machine $(1, TOU|states,\ speeds,\ q_j|TEC)$. We proposed two mathematical models for this problem and three algorithms (one heuristic, one genetic and one memetic algorithms) to solve the medium and large size instances of this NP-hard problem. The complexity of two sub-problems is also analyzed. For this purpose, the scheduling problem of several jobs with different energy consumptions and on a predetermined sequence are studied into two cases: uniform-speed machine and speed-scalable machine. By using dynamic programming approaches and finite graphs, it is proved that the problem $(1, TOU|states,\ sequence,\ q_j|TEC)$ is polynomial, and the problem $1, TOU|states,\ sequence,\ speeds,\ q_j|TEC$ is pseudo-polynomial. The contributions of this thesis in terms of complexity analysis are presented in two figures (5.1 and 5.2). Figure 5.1 presents the results for different variants of $1, c_t|states|TEC$ problem, and Figure 5.2 presents the results for different variants of $1, TOU|states,\ q_j|TEC$ problem.

As the future works of this study, for the short-term perspectives, we are interested to improve the performance of the presented algorithms for $1, TOU|states,\ speeds,\ q_j|TEC$ problem and develop other exact methods for this problem. For example, it could be interesting to provide some dominance rules to improve the performance of branch and bound method for this problem. Furthermore, we search to propose some other local search method to improve the performance of the proposed memetic algorithm. Also, we want to generalize the model to consider the release dates and the due dates for each job. The consideration of multi-objective problems, which

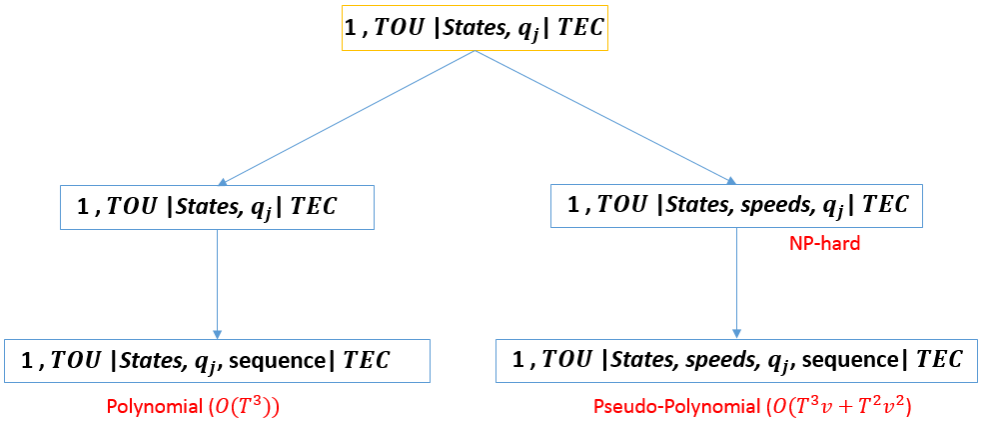

Figure 5.2 – Our contributions for different variants of $1, TOU|states,\ q_j|TEC$ problem

optimize the total energy consumption costs in addition to a traditional scheduling objective like makespan seems to be relevant and promising.

With regard to long-term perspectives, it must be mention that the basic idea of this thesis was to introduce some energy and ecological considerations in the scheduling problem of a single machine manufacturing system. A manufacturer is linked to a producer (supplier) of energy by a contract specifying the change in the unit of energy price over time slots, same as the variation of the power. Therefore, the feasibility of production plans must be considered with the available energy capacity.

Indeed, the energy market will evolve in the years ahead, especially, for the clean energy sources such as solar or wind power. This can result in the number of expanding research interests on this subject. A manufacturer will have the opportunity to link up with several suppliers. Some suppliers offer clean energy, but with uncertainty in the quantity and the availability. Other suppliers offer less clean energy and more taxed but with guarantees of availability. Therefore, we plan to use these early works to offer some decision-making method which allow the manufacturer to choose it's energy sources policy by varying its suppliers and minimizing its total energy consumption costs over a given horizon.

Part II

French version

French Summary

Outline of the current chapter

Introduction

Au cours des dernières années, l'évolution économique et sociétale a entraîné une augmentation rapide de la consommation d'énergie et le risque pénurie d'énergie est devenue un obstacle à la croissance économique dans de nombreux pays ([1]). Parallèlement, les émissions de CO2 générées par l'utilisation de l'énergie sont devenues l'un des principaux facteurs du changement climatique mondial et de l'effet de serre. Depuis la révolution industrielle, le taux de gaz à effet de serre a augmenté de presque 70 % (entre 1970 et 2004). Le secteur industriel est le plus grand consommateur d'énergie dans la plupart des pays, il est donc important de se focaliser sur ce dernier pour essayer de réduire la consommation d'énergie et par conséquent les émissions de gaz à effet de serre.

En outre, les prix de l'électricité dans presque tous les pays industrialisés, n'ont cessé d'augmenter. Cette situation résulte principalement des taxes et des droits de douane visant à soutenir l'intégration des énergies renouvelables et de l'abandon de l'électricité à bas prix produite à partir de l'énergie nucléaire. En conséquence, la part des coûts énergétiques par rapport aux coûts de production augmente, ce qui se traduit par une compétitivité moindre par rapport aux pays où les prix de l'électricité augmentent plus au moins lentement.
En raison des fluctuations croissantes des prix de l'électricité, il est possible de réaliser des économies d'énergie sans investissements à forte intensité de capital grâce à des méthodes d'organisation spécifiques. Ces méthodes tentent de traiter les commandes énergivores en période de bas prix et les commandes peu énergivores en période de prix élevés. Ces questions ont encouragé de nombreux chercheurs du monde entier à travailler sur l'amélioration de l'efficacité énergétique des systèmes de production dans le but de réduire à la fois les coûts de production et de réduire l'empreinte écologique.

Il existe dans la littérature de nombreux ouvrages qui intègrent l'aspect écologique dans les problèmes de décision. Ils peuvent être distingués en fonction du niveau de décision considéré : stratégique, tactique et opérationnel. Au niveau stratégique, les travaux sont généralement liés à des problèmes de chaîne d'approvisionnement pour lesquels les chercheurs veulent déterminer l'emplacement des sites, les capacités et le nombre de centres de distribution, afin de minimiser

les coûts totaux et les émissions de gaz à effet de serre. Au niveau tactique, les travaux traitent de l'optimisation de la planification et de la gestion des ressources pour gérer la consommation d'énergie afin de limiter les émissions de gaz à effet de serre et les déchets. Au niveau opérationnel, les études de recherche abordent les problèmes d'ordonnancement pour maximiser la performance d'une ligne de production en tenant compte des facteurs énergétiques.

Dans cette thèse, nous nous intéressons au niveau opérationnel. Une large étude des travaux antérieurs montre que la consommation d'énergie d'un système de fabrication peut être minimisée à trois niveaux : machine, produit et système. Notez qu'au niveau de la machine et du produit, d'énormes investissements financiers et du temps sont nécessaires pour concevoir la ou les nouvelles machines ou le ou les nouveaux produits qui consomment moins d'énergie que la précédente. Au niveau du système, les fabricants peuvent réduire la consommation d'énergie de leur système en utilisant plusieurs modèles de décision et techniques d'optimisation pour gérer le plan de production.

Dans l'ensemble de cette thèse, le niveau système est abordé pour fournir des méthodes de réduction de la consommation d'énergie pour un système de production. Nous nous sommes concentrés sur les problèmes d'ordonnancement de la production qui intègrent les consommations d'énergie.
C'est pourquoi, dans ce qui suit, nous présentons tout d'abord une brève introduction sur les différents types de problèmes d'ordonnancement. Ensuite, nous nous focalisons sur les problèmes d'ordonnancement à une seule machine avec aspects énergétiques et une revue complète des études précédentes est présentée.

État de l'art

Dans cette étude, nous nous concentrons sur les problèmes d'ordonnancement existants dans un système de fabrication pour améliorer l'efficacité de la production et réduire les coûts.
Dans un système de fabrication, lorsque les ordres sont lancés, ils doivent être traduits dans les jobs avec les dates d'échéance associées. Ces ordres de fabrication doivent souvent être traités en

séquence par les machines dans un poste de travail. Les problèmes d'ordonnancement se posent dans ce contexte.
Dans la littérature, on trouve différentes définitions et classifications des problèmes d'ordonnancement. Une des classifications les plus répondues pour les problèmes d'ordonnancement classique est présentée dans [13]. Les auteurs ont introduit la notation à 3 champs qui s'appelle la notation de Graham ($\alpha|\beta|\gamma$). Dans cette notation, le premier champ (α) définit l'environnement machine. Le deuxième champ (β) décrit les caractéristiques des tâches et les contraintes du problème d'ordonnancement. Le troisième champ (γ) dans la notation de Graham fournit la fonction objective à optimiser ([10]).

Une étude assez exhaustive des études antérieures sur l'ordonnancement de la production et les problèmes de consommation d'énergie montre qu'il existe différentes méthodes pour intégrer le concept de consommation d'énergie dans l'ordonnancement de la production. Par exemple, [16] une revue de la littérature des modèles d'aide à la décision pour la planification avec aspects énergétiques. [2] a présenté plusieurs options sur la façon dont les concepts d'optimisation à l'échelle de l'entreprise peuvent intégrer la gestion et la planification énergétiques. [3] a étudié des approches d'ordonnancement qui visent à améliorer l'efficacité énergétique. Ils ont classé la littérature en fonction de trois aspects : la couverture énergétique, l'offre énergétique et la demande d'énergie.

L'efficacité énergétique d'un système de fabrication peut être étudiée de différentes manières, comme une tentative de diminuer la valeur de la consommation d'énergie, de réduire le coût de la consommation d'énergie (coût opérationnel), et de tenir compte des contraintes énergétiques. D'après l'analyse documentaire présentée, parmi 90 articles, il n'y a que 24 papiers qui ont étudié la consommation d'énergie et les facteurs de coût de l'énergie en même temps. A notre connaissance, très peu de publications (7 articles sur 90 recherches) traitent de l'efficacité énergétique d'un système à une seule machine et plusieurs états avec le coût de l'électricité en fonction du temps.

Dans cette thèse, nous sommes intéressés à combler ce manque dans la littérature. Pour ce faire, un seul système machine avec trois états principaux comme ON (traitement), Idle, et OFF.

Deux états de transition entre les états ON et OFF sont aussi pris en compte. Notre objectif est, dans un premier temps, d'analyser la complexité de ce type de problèmes et enfin de proposer des méthodes d'optimisation pour les résoudre.

Parmi les articles étudiés, nous venons de trouver un article ([57]) qui étudiait la minimisation des coûts énergétiques totaux en plus de la consommation d'énergie dépendant de l'état et de l'emploi. De plus, il n'existe qu'un seul document ([76]) qui étudie la minimisation des coûts énergétiques totaux en plus de la consommation d'énergie dépendante de l'état et dépendante de l'état. Il n'existe aucun travail antérieur sur la consommation d'énergie dépendante de l'état, de l'emploi et de la vitesse en tout. Par conséquent, plusieurs types de ce problème avec des hypothèses différentes sont étudiés dans les sections suivantes.

Problème d'ordonnancement à plusieurs états d'une seule machine économe en énergie avec séquence fixe

Cette section traite de l'efficacité énergétique d'un problème d'ordonnancement d'une seule machine à plusieurs états lorsque la séquence des tâches est fixe, et les tarifs d'électricité TOU sont considérés pour chaque période sont considérés. Ainsi, les coûts énergétiques varient d'une période à l'autre et la consommation d'énergie des états de la machine est différente. Par conséquent, dans ce problème, le problème d'ordonnancement cherche un ordonnancement optimal pour allouer les états à faible consommation aux heures de pointe, et les états à consommation élevée aux heures creuses.

Définition du problème

Un problème d'ordonnancement non préemptif de n tâches non-préemptive dans un ordre prédéterminé sur une seule machine est étudié dans [61]. La machine en question peut évoluer entre trois états différents : ON (traitement), OFF ou Idle. Lorsque la machine est à l'état OFF, un nombre fixe de périodes doit s'écouler jusqu'à ce que la machine soit prête à traiter un travail une tâche (i.e. Ton (β_1)). De même, lorsque la machine est à l'état ON, un nombre fixe différent

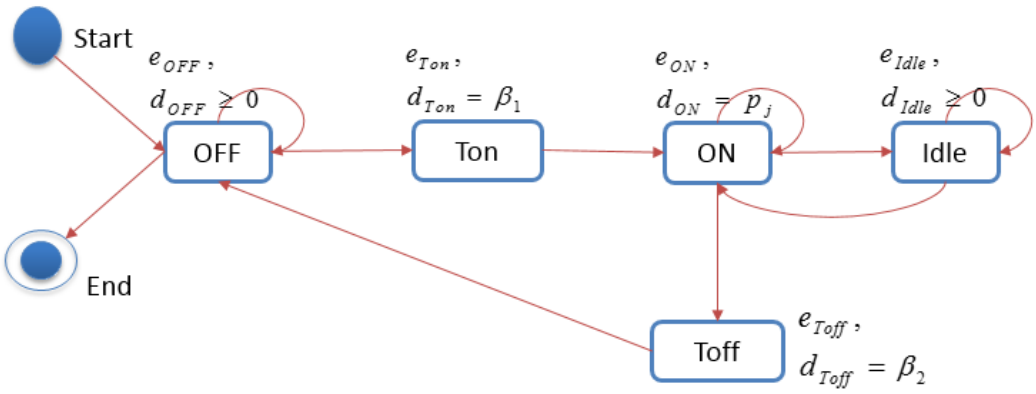

FIGURE 5.3 – Les états et transitions considérés pour la machine.

de périodes doit s'écouler jusqu'à ce qu'elle soit réellement à l'état OFF (i.e. Toff (β_2)). Comme le montre la Figure 5.3, il est supposé que :

- L'horizon de production se compose de T périodes.
- Chaque période t ($0 \leq t \leq t \leq T$) est caractérisée par un prix de l'électricité (c_t).
- n tâches avec des temps de traitement différents (p_j) doivent être traités par la machine.
- Chaque état de la machine est caractérisé par une consommation d'énergie, où leurs valeurs sont l'entrée du problème ($e_{On}, e_{Off}, e_{Off}, e_{Idle}, e_{Ton}, e_{Toff}$).
- La machine est en état d'arrêt à la période initiale ($t = 0$) et la période finale ($t = T$).

Formulation mathématique

Un modèle mathématique LP a été proposé par les auteurs, pour trouver la solution optimale (le planning le moins cher en termes de coûts totaux de consommation d'énergie) pendant un horizon de production, en prenant des décisions au niveau de la machine, avec une séquence de travail fixe. Leur modèle détermine l'allocation optimale du traitement des tâches, le temps d'inactivité, l'état d'arrêt et les temps de transition pour la mise en marche et l'arrêt. Dans cette section, nous avons d'abord proposé un modèle amélioré pour ce problème, puis nous présentons une nouvelle approche de programmation dynamique pour analyser sa complexité.

Dans le modèle initial proposé par Sharouf et al.([61]), les auteurs utilisent deux variables pour définir la position des tâches dans chaque période (une variable ($y_{j,t}$) pour identifier le moment où la tâche commence à être traité et une autre ($x_{j,t}$) pour déterminer les périodes durant lesquelles la tâche sera traité).

$$x_{j,t} = \begin{cases} 1 & ; \text{ Si tâche } j = 1, ..., n \text{ est traité pendant la période } t \\ 0 & ; \text{ Sinon} \end{cases}$$

$$y_{j,t} = \begin{cases} 1 & ; \text{ Si tâche } j = 1, ..., n \text{ commence à être traité en période } t \\ 0 & ; \text{ Sinon} \end{cases}$$

L'une des hypothèses de cette étude est que la préemption des jobs n'est pas autorisée. Cela signifie que lorsqu'une tâche commence à être traité, la machine doit continuer le traitement jusqu'à ce qu'elle soit terminée. Ainsi, nous ne pouvons utiliser qu'une seule variable pour définir la situation de la tâche. Pour ce faire, au lieu d'utiliser deux variables ($x_{j,t}$ et $y_{j,t}$), le modèle est défini en fonction d'une seule variable de décision ($y_{j,t}$).

$$y_{j,t} = \begin{cases} 1 & ; \text{ Si tâche } j = 1, ..., n \text{ est effectué dans la période } t \text{ par la machine} \\ 0 & ; \text{ Sinon} \end{cases}$$

Puisque j peut prendre des valeurs différentes de n de 1 à n, et t peut prendre des valeurs différentes de $T + 1$ de 0 à T, ce changement réduira $n * (T + 1)$ du nombre de variables. Cette valeur sera montant impressionnant pour un grand nombre des tâche et de périodes.

Pour explorer les différences entre ces deux modèles mathématiques, en nombre de variables et de contraintes ainsi qu'en temps de calcul, les deux modèles ont d'abord été testés à partir des quatre instances présentées par [61]. Les résultats montrent que le nombre de variables et de contraintes ainsi que le temps de calcul dans le modèle amélioré sont réduits par rapport au modèle de base. Ensuite, trois stratégies différentes et de nombreux exemples basés sur chaque stratégie sont considérés. Toutes les instances sont résolues avec les deux modèles à l'aide du solveur CPLEX et sur le même ordinateur. Il faut mentionner que le temps de calcul pour toutes les instances est limité à une heure (3600 s).

En conséquence, dans le nouveau modèle, en éliminant une variable et en réécrivant certaines contraintes, non seulement le temps de calcul et le nombre de variables sont significativement réduits (en moyenne 78,64 % et 39,62 % du temps de calcul et du nombre de variables) mais aussi le nombre de contraintes est diminué ce qui peut être très intéressant surtout pour un grand nombre de tâches. A titre d'exemple, dans les temps impartis de 3600 s, le modèle de base est capable de résoudre de maniéré optimale des instances de taille (30,107) alors que notre modèle est capable de traiter des instances de tailles plus importantes (jusqu'à 45 tâches et,153 périodes).

Analyse de complexité

Dans cette section, nous prouvons qu'il est possible de résoudre de manière optimale ce problème avec un algorithme polynomial basé sur une approche de programmation dynamique. Pour cela, un graphe fini dont les dimensions (nombre de sommets et de nœuds) dépendent des temps de traitement totaux et du nombre total de périodes est utilisé pour modéliser le problème. Puis nous avons utilisé une approche de programmation dynamique pour trouver la solution optimale et nous avons discuté de la complexité de la recherche du chemin le plus court dans ce graph.

Dans le problème considéré, caractérisé par un nombre fixe de périodes (T), un nombre minimum de périodes est requis pour accomplir les tâches nécessaires. Ces tâches comprennent la mise en marche et l'arrêt de la machine pendant au moins une période et l'exécution de toutes les tâches. Le nombre de périodes requises pour exécuter ces dernières (P) est égal à la somme des temps de traitement ($P = \sum_{j=1}^{n} p_j$). De plus, au moins $\beta_1 + \beta_2 + 1$ sont nécessaires pour les états d'allumage initial, d'extinction finale et d'extinction finale (ne serait-ce qu'une fois allumés et éteints pour être considérés). Pendant les périodes restantes (différence entre le nombre total de périodes et le nombre minimal requis de périodes), la machine doit se trouver dans des états sans traitement, c'est-à-dire dans les états OFF initial, OFF au milieu, OFF final et inactif. Notez que chaque état middle-OFF se compose d'une séquence de Toff, OFF pendant au moins une période, et Ton. Soit x indique le nombre de périodes supplémentaires, donc :

$$x = T - P - (\beta_1 + \beta_2 + 1) \tag{5.1}$$

Selon l'objectif du problème, ces périodes x peuvent être attribuées à une combinaison d'états OFF initiaux ou finaux, d'états inactifs entre les états ON et d'états OFF intermédiaires.
Dans cette approche, un graphe composé de plusieurs niveaux décisionnels est considéré pour modéliser le problème. Chaque niveau représente une période de l'horizon. Ainsi, le graphe se compose des niveaux de décision $T + 1$ $(0 \leq l \leq l \leq T)$.
Pour chaque niveau de décision, considérons H_l composé des nœuds possibles pour le niveau l, qui correspond aux différents états de la machine. Par conséquent, chaque nœud de H_l est caractérisé par le nombre cumulé d'unités de production (k) de la période 0 à l. Puisque les états initial et final de la machine sont considérés comme étant l'état OFF, $H_0 = \{I\}$ et $H_T = \{F\}$, où I représente l'état initial de la machine, et F représente l'état final de la machine après avoir traité toutes les tâches (P).
Les périodes les plus anciennes et les plus récentes possibles (ou les niveaux dans la définition de ce graphe) pour n'importe quel nœud k dépendent de la valeur x du problème donné. Ces niveaux peuvent être caractérisés par l'intervalle $\tau_k =$
$l_{min(k)}, \cdots, l_{max(k)}\}$ dans l'horizon de planification. Par conséquent, la première étape de cette approche de programmation dynamique est de mettre tous les nœuds $(k \in \{I, 1, \cdots, P, F\})$ dans le graphe en utilisant leurs intervalles de temps associés (τ_k).

La deuxième étape consiste à représenter les nœuds du graphe. Dans cette approche, les nœuds du graphe représentent les transitions possibles entre deux nœuds (nœud (k, l) et nœud (k', l')).
La troisième étape consiste à valoriser les nœuds. Dans cette approche, les nœuds sont évalués par les coûts totaux de consommation d'énergie pour effectuer la transition correspondante : $(Ev_{(k,l)-(k',l',l')};\ \forall\ k \in H_l,\ k' \in H_{l'},\ l' \geq l + 1)$.

Le nombre total de sommets (V) et les arcs (E) du graphe proposé sont :

$$|V| = P \times (x + 1) + 2 \cong TP \tag{5.2}$$

period	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cost	0	3	2	5	4	2	3	4	7	2	5	4	6	1	3	2

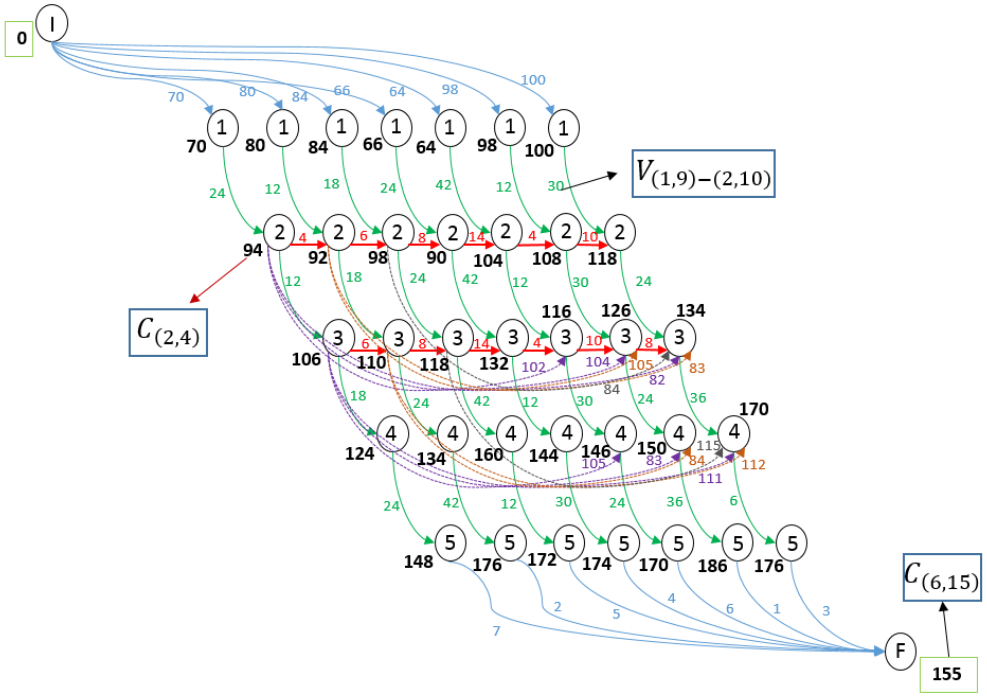

FIGURE 5.4 – Représentation graphe du problème de séquence fixe

$$|E| \cong T^2 P \tag{5.3}$$

Pour illustrer cette méthode de construction graphe, nous considérons un exemple avec $P = 5, T = 15, \beta_1 = 2, \beta_2 = 1, x = 6$, et un prix de l'énergie variable dans chaque période (c_t). Le graphe correspondant se compose de 37 sommets et de 66 arcs (voir Fig. 5.4).

La quatrième étape de cette approche consiste à trouver la solution optimale du problème. Selon l'approche de modélisation graphe, chaque chemin du nœud I du niveau 0 au nœud F du niveau T représente une solution réalisable du problème et son poids représente les coûts totaux de consommation d'énergie. Puisque l'objectif est de minimiser les coûts énergétiques totaux, le chemin le plus court qui commence au noeud $(I, 0)$ et se termine au noeud (F, T) représente la solution optimale du problème.

Toutes les valeurs marginales du graphe sont positives dans cette approche. Ainsi, l'algorithme de Dijkestra, qui est l'un des algorithmes les plus efficaces pour trouver le chemin le plus court entre le nœud source et tous les autres nœuds d'un graphe, est applicable pour notre approche.

Après avoir décrit la nouvelle approche de formulation, nous concluons que cette approche est capable de modéliser toutes les solutions possibles du problème en un temps polynomial. Ensuite, nous voulons prouver que la solution optimale de ce problème peut être obtenue dans un temps polynomial aussi. Par conséquent, la complexité de l'algorithme de Dijkstra pour ce problème est évaluée.

Selon [77], la pire implémentation de l'algorithme de Dijkstra est basée sur une file d'attente basée sur une priorité, a une complexité de $O(|E| + |V| \log |V|)$ (où $|E|$ est le nombre de nœuds et $|V|$ est le nombre d'arcs). Par conséquent, la complexité de cet algorithme pour l'approche de programmation dynamique est donnée par :

$$O(T^2P + TP \log TP) = O(T^2P + TP \log T + TP \log P) \cong O(T^2P) \tag{5.4}$$

Puisque la plus grande valeur possible de P est T (analyse du pire des cas), le problème considéré est un polynôme de degré 3 ou un problème polynomial cubique ($O(T^3)$). Notez que, dans cette étude, l'algorithme de Dijkstra est choisi, alors que dans la littérature il existe d'autres algorithmes qui peuvent trouver le chemin le plus court plus rapidement. Mais, puisque notre objectif est d'analyser la complexité de ce problème, nous n'avons pas vérifié les autres algorithmes. Considérant qu'il est possible d'optimiser le temps de calcul de cette approche en utilisant un autre algorithme.

Après avoir analysé la complexité de ce problème, il est très intéressant d'analyser la complexité de la version générale de ce problème sans séquence fixe des tâches $(1, TOU|états|TEC)$. A cette fin, dans la section suivante, la complexité de ce problème et d'autres sous-problèmes est étudiée.

Problème d'ordonnancement à plusieurs états d'une seule machine économe en énergie : version généralisée

L'étude de Shrouf et al. [61] et la section précédente traitent du problème dans lequel la séquence des tâches est prédéterminée. Ainsi, les modèles présentés ne font que spécifier l'état optimal de la machine dans chaque période en fonction de la minimisation du coût total de l'électricité.
Néanmoins, dans la plupart des industries manufacturières, trouver la séquence optimale des tâches est une question primordiale. A notre connaissance, il n'existe aucune étude qui considère le problème d'efficacité énergétique tel que défini ici pour trouver la séquence optimale des tâches combinée à l'ordonnancement des états de la machine. Par conséquent, dans cette section, une généralisation du problème précédent est proposée pour trouver les séquences optimales des états de la machine et des tâches simultanément.
Nous proposons une modélisation mathématique ainsi que plusieurs méthodes d'optimisation sont proposées pour résoudre ce problème. Nous présentons également une analyse de la complexité du plusieurs variantes de ce problème.

Présentation du problème

Cette section traite du problème d'ordonnancement de plusieurs travaux sur une seule machine multi-états sans séquence fixe $(1, TOU|états|TEC)$. Ainsi, nous conservons toutes les hypothèses et contraintes de la section II, et nous relâchons simplement la contrainte de la séquence prédéterminée. De cette façon, la machine peut traiter les tâches de manière optimisée afin de minimisation des coûts de consommation de l'énergie.

Analyse de complexité

L'une des premières étapes de l'étude d'un nouveau problème consiste à analyser sa complexité pour savoir s'il s'agit d'un problème NP-dur ou d'un problème polynomial. Nous avons prouvé sa complexité (NP-difficile) du problème $(1, TOU|états|TEC)$ en utilisant une réduction en un problème de décision à 3-PARTITION.

Analyse de complexité des sous-problèmes

Puisque la version générale du problème est NP-hard, nous étudions la complexité de certains sous problèmes. Pour définir les sous-problèmes, nous avons retenu les hypothèses principales comme problème général pour tous les sous-problèmes et nous n'en avons changé qu'une seule pour chaque sous-problème.

Pour ce faire, on examine tout d'abord la complexité du problème en considérant une évolution régulière des prix de l'énergie. Ensuite, nous avons étudié le problème lorsque la préemption des tâches est permise et, enfin le problème des mêmes temps de traitement pour les tâches.
Comme résultats, nous avons prouvé que les problèmes avec un prix constant de l'énergie $(1, c_t = c|états|TEC)$ et les problèmes avec prix croissants ou décroissants sont polynomiaux.

Après avoir analysé l'effet des coûts énergétiques sur la complexité de ce problème, nous avons étudié la complexité du cas préemptif. Pour ce faire, la même approche par programmation dynamique présentée dans la section 2.4.1 est utilisée. La différence entre l'approche de la version à séquence fixe de cette étude et sa version à préemption réside dans le nombre de nœuds. Pour la version avec séquence fixe, les états non traitants peuvent être répartis entre deux jobs de traitement, ce qui signifie que les nœuds correspondants aux états intermédiaires et inactifs peuvent apparaître entre deux jobs lorsque le premier job est terminé et que le second n'est pas lancé. Ainsi, le graphe associé au cas avec préemption a plus d'arcs que le graphe associé à la version avec séquence fixe, alors que les deux ont le même nombre de nœuds.

$$|V| = (P+2) \times (x+1) \cong TP \tag{5.5}$$

$$|E| = ((P+2) \times x) + ((P+1) \times (x+1)) + ((P-1) \times [\frac{(x-(\beta_1+\beta_2)) \times (x-(\beta_1+\beta_2)+1)}{2}]) \tag{5.6}$$

$$\Rightarrow \quad |E| \cong T^2 P \tag{5.7}$$

Par conséquent, la complexité de l'algorithme de Dijkestra pour l'approche de programmation dynamique présentée est donnée comme suit :

$$O(T^2P + TP\log TP) \cong O(T^2P) \tag{5.8}$$

Comme la plus grande valeur pour P est T, donc, tous ces résultats prouvent que dans le pire des cas, la complexité de ce problème est donnée par un polynôme de degré 3.

La dernière analyse de cas des sous-problèmes de $(1, TOU|états|TEC)$ concerne les problèmes avec les mêmes délais de traitement pour les emplois $(1, TOU|états, p_j = p|TEC)$. Pour programmer les états de la machine au cours d'une période donnée, deux facteurs importants sont la consommation d'énergie de la machine pendant chaque état et l'unité du prix de l'énergie pendant chaque période. Dans le problème général $(1, TOU|états|TEC)$, on considère que la consommation d'énergie de la machine pendant les états ON est indépendante du travail traité. Ainsi, le seul paramètre qui a causé une préférence entre les jobs est leur temps de traitement. Ainsi, lorsque les travaux ont les mêmes temps de traitement, la machine consomme la même quantité d'énergie pour les réaliser. Considérons que chaque travail nécessite des périodes p à traiter $(p_j = p)$, et que la machine consomme e_{ON} unit de l'énergie par période pendant les états ON. Ainsi, la machine consomme $e_{ON} * p$ unités d'énergie pour traiter chaque tâche.

Dans cette configuration, il n'y a pas de différence entre les différentes séquences des tâches $\{(e_{ON} * p) - (e_{ON} * p) - (e_{ON} * p) - (e_{ON} * p) - \cdots - (e_{ON} * p)$Ċela signifie que, lorsque la machine est à l'état ON, n'importe quel tâche être traitée sans affecter l'optimalité de la solution. Ainsi, le problème $1, TOU|états, p_j = p|TEC$, peut être considéré comme un problème de séquence fixe dans lequel chaque tâche a besoin de p temps unitaires comme temps de traitement $(1, TOU|états, séquence, p_j = p|TEC)$. La version générale de ce problème $(1, TOU|états, séquence$

$|TEC)$ examinée précédemment qui est polynomial. Par conséquent, le sous-problème $(1, TOU|$ $états, p_j = p|TEC)$ est aussi polynomial.

Méthodes de résolution du problème $(1, TOU|états|TEC)$

Habituellement, les méthodes exactes ne peuvent pas trouver la solution optimale pour les cas de grande taille d'un problème NP-difficile dans un délai raisonnable. Ainsi, pour le problème présenté dans cette section qui est NP-hard $(1, TOU|états|TEC)$, nous proposons des algorithmes

heuristiques et méta-heuristiques afin de résoudre les instances de grandes tailles. Il s'agit d'une heuristique dédiée et d'un algorithme génétique.

Comme nous l'avons mentionné ci-avant, la condition nécessaire pour avoir les solutions réalisables à un problème avec un nombre fixe de périodes est que le nombre de périodes soit supérieur ou égal à la durée totale tes temps de traitement des tâches en plus du nombre de périodes nécessaires pour les transitions entre les états de la machine. Ces périodes sont nécessaires pour mettre la machine en marche, effectuer toutes les tâches complètement et l'éteindre ensuite. La différence entre ces deux nombres indique le nombre de périodes supplémentaires qui peuvent être calculées par équation 5.1 et notée x. Le principe de l'heuristique proposée sont basés sur l'affectation des états (Ton, OFF, Idle et Toff) de la machine pendant ces périodes supplémentaires. Les périodes supplémentaires peuvent être placées n'importe où sur un horizon de production (au début, à la fin ou au milieu). Ainsi, notre algorithme est divisé en deux étapes générales (exploration à partir du début de l'horizon et exploration en commençant par la fin de l'horizon) pour spécifier l'état de la machine période par période et considérer toutes les possibilités.

A chaque étape, l'algorithme examine $(x + 1)$ situations différentes (de 0 à x) pour allumer la machine au début ou l'éteindre à la fin de l'horizon, et sélectionne la celle qui minimise le coût parmi différentes solutions $(x + 1)$. De cette façon, pour chaque solution, des périodes supplémentaires seront allouées à l'état OFF à la première étape et ensuite, l'algorithme décide pour le reste (entre les états OFF ou Idle) pour les autres périodes. Enfin, la solution est celle qui minimise la fonction-objectif parmi toutes les solutions explorées. Ainsi, pour obtenir la meilleure solution, des solutions différentes de $2 *(x + 1)$ doivent être considérées et comparées les unes avec autres. Cette méthode (déterminer le nombre de solutions de contrôle) permet de réduire la dépendance de l'algorithme au nombre de tâches et de périodes (taille du problème) et d'augmenter son efficacité.

Un algorithme génétique est également présenté pour examiner la possibilité d'améliorer les solutions obtenues par l'heuristique.

Puisque, chaque solution de ce problème revient à déterminer l'état de la machine pour chacune des T périodes. Ainsi, dans cet algorithme génétique, chaque chromosome est représenté

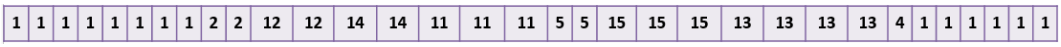

FIGURE 5.5 – Le chromosome correspondant à la solution obtenue sur la figure 3.2

par $T + 1$ gènes et chaque gène identifie l'état de la machine dans pour chacune des périodes. De plus, pour distinguer les différents états, les nombres de 1, 2, 4 et 5 sont supposés représenter respectivement les états OFF, Ton, Toff et Idle.
En outre, un nombre entier supérieur à 10 ($w > 10$), représente que la machine est dans l'état ON et traite le $(w - 10)th$ job. Le chromosome correspondant à la solution présentée à la figure 3.2 est représenté à la figure 5.5.

Habituellement, l'algorithme génétique (AG) commence par une population initiale générée au hasard. L'AG proposée utilise une population initiale qui sera générée sur la base de notre algorithme heuristique proposé avec une population de 300 personnes. Ainsi, pour chaque individu, il faut tout d'abord sélectionner au hasard la période de mise en marche de la machine. Ensuite, le numéro du travail à effectuer sera choisi au hasard. Une fois le travail terminé, l'état de la machine doit être sélectionné arbitrairement entre les états ON, Toff et Idle. Ces procédures doivent être poursuivies pour traiter tous les jobs jusqu'à la dernière période. Enfin, la valeur objective de la solution sera calculée en fonction de l'aptitude à classer la qualité du chromosome généré.
Pour compléter la procédure de l'AG, nous utilisons les mêmes opérateurs que l'étude existante dans la littérature ([61]) pour les procédures de sélection des parents, de crossover et de mutation. Par conséquent, la sélection de la roue de roulette est sélectionné pour la sélection des parents et un seul point de croisement ainsi que la méthode de swap pour l'opérateur de mutation, sont considérés pour produire deux nouveaux enfants.

Afin d'évaluer la performance de ces deux algorithmes proposés, ils ont été codés en langage $C++$ en utilisant Visual Studio 2015. Ces derniers ont été testés sur plusieurs instances générées aléatoirement.

Les résultats obtenus démontrent que les algorithmes que nous proposons fournissent les

solutions optimales pour certains exemples et des solutions presque optimales pour d'autres en quelques secondes (l'écart de 2,2% est atteint en moyenne par HA et 1,82% par GA en comparaison avec la valeur de la fonction-objectif de la solution exacte pour les instances de petite taille). De plus, parmi toutes les expérimentations, l'écart le plus important, soit 5,67 % et 7,31 %, est obtenu respectivement par l'AP et l'AG. Dans les cas de grande taille, la fourchette de variation des écarts entre l'AH et l'AG est inférieure à 3,0 % et, en moyenne, elle est approximativement égale à 1,0 %.

Bornes inférieures pour le problème de $(1, TOU|\textbf{é}tats|TEC)$

Un outil habituel pour évaluer les performances des méthodes approximatives pour un problème NP-dur est d'obtenir quelques limites inférieures. C'est pourquoi, dans cette thèse, nous avons également proposé quatre limites inférieures pour le problème de $(1{,}00TOU|états|TEC)$. Pour évaluer l'efficacité des limites inférieures proposées dans la présente étude, plusieurs cas générés au hasard sont examinés. Les résultats montrent qu'entre LB_1, LB_2, et LB_3, dans tous les cas LB_2 propose une meilleure borne. Parmi ces limites inférieures, LB_4 qui est la solution optimale obtenue du cas préventif de ce problème par CPLEX, trouve la solution la plus proche de la solution optimale. L'ordre de classement de ces limites inférieures est le suivant :

$Gap_{LB_4} < Gap_{LB_2} < Gap_{LB_1} < Gap_{LB_3}$

De plus, une analyse de la variance (ANOVA) avec un niveau de confiance de 95% a été effectuée à l'aide du logiciel Minitab.17 pour vérifier la validité statistique des résultats (Fig. 5.6). Comme on peut le voir dans cette figure, pour chaque taille de problème, l'intervalle des écarts pour toutes les limites inférieures proposées ($LB_1,\ LB_2,\ LB_3,\ LB_4$) est présenté. Dans tous les cas, LB_4 a l'intervalle minimum des écarts.

La section suivante traite d'un problème d'ordonnancement d'une seule machine à plusieurs états lorsque les taches consomment différentes quantités d'énergie. Pour cela, deux versions de ce problème avec une vitesse uniforme et une machine à vitesse variable sont étudiées.

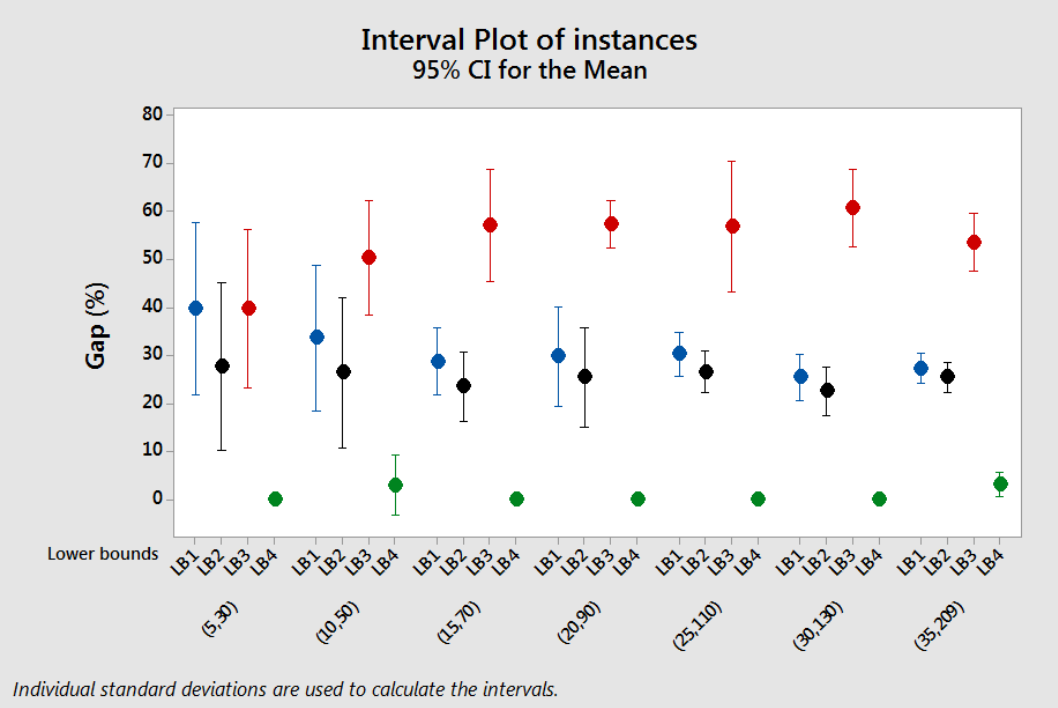

FIGURE 5.6 – Comparaison des performances des bornes inférieures avec les solutions optimales obtenues par CPLEX

5 Problème d'ordonnancement avec une machine ayant plusieurs états et à plusieurs vitesses

En réalité, pour certaines industries manufacturières, le traitement de différentes tâches avec la même machine nécessite des consommations d'énergie différentes. Elle peut provenir des tâches traitées ou de la vitesse de traitement de ces dernières. C'est pourquoi, dans cette section, nous nous intéressons la version plus générale du problème avec des états multiples et des vitesses multiples. En outre, la complexité de plusieurs sous-problèmes est également analysée.

Présentation du problème

Cette section traite du problème d'ordonnancement de plusieurs tâches sur une seule machine ayant des états multiples et pouvant exécuter chacune des tâches avec plusieurs vitesses $(1, TOU|états, vitesses, q_j|TEC)$. De plus, pour le problème étudié dans cette section, la consommation d'énergie de la machine à l'état ON dépend de deux facteurs : la tâche à traiter et

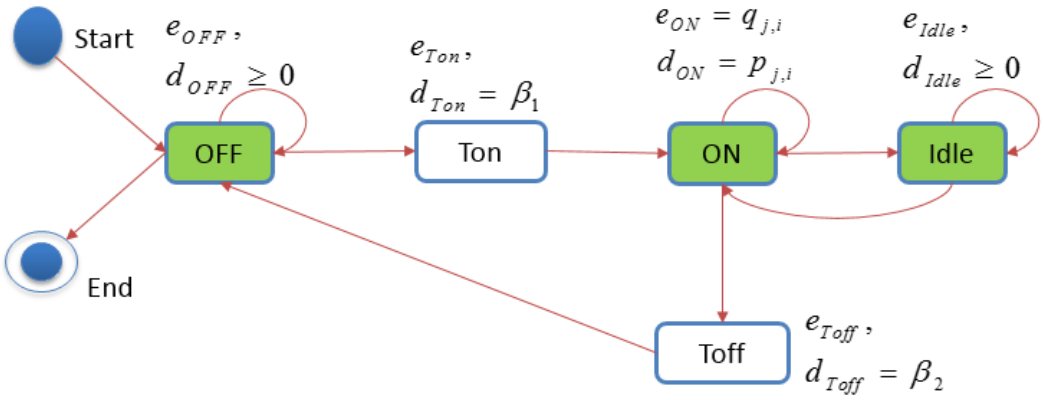

FIGURE 5.7 – Les états et transitions considérés pour la machine.

la vitesse du traitement de la machine. Ainsi, il y a différentes possibilités pour le temps de traitement de chaque tâche et la valeur énergétique consommée par la machine. Cela signifie que pour chaque job $j = 1, ..., n$, v_j vitesses de traitement sont possibles, il y a différentes valeurs pour le temps de traitement comme $P_j = \{p_{j,1}, \cdots, p_{j,v_j}\$$, et pour chaque $p_{j,i}$, une consommation énergétique correspondante $q_{j,i}$ est associée. $Q_j = \{q_{j,1}, \cdots, q_{j,v_j}\}$ est l'ensemble des différentes consommations énergétiques du job $j = 1, ..., n$. En se basant sur le fait que le traitement plus rapide d'un travail prend moins de temps et consomme plus d'unités d'énergie, les relations suivantes sont prises en compte :

$$p_{j,1} > p_{j,2} > \cdots > p_{j,v_j} \quad ;\forall j \in \{1, \cdots, n\} \tag{5.9}$$

$$q_{j,1} < q_{j,2} < \cdots < q_{j,v_j} \quad ;\forall j \in \{1, \cdots, n\} \tag{5.10}$$

L'objectif de ce problème est de trouver l'ordonnancement de la production le plus économique en termes de coûts de consommation d'énergie sur l'ensemble de l'horizon de temps. Les états machine et les transitions possibles ainsi que les consommations d'énergie de chaque état sont illustrés dans la Figure5.7.

Comme première contribution, deux nouveaux modèles mathématiques sont proposés pour ce problème. La performance de ces deux modèles mathématiques a été examinée à travers plusieurs instances générées aléatoirement. Les résultats montrent que le deuxième modèle est plus rapide que le premier (en moyenne, il faut 271.87 s pour le premier modèle, et 1.80 s pour le second), et il a diminué d'environ 64 % en nombre de contraintes et 1.3% en nombre de variables.

Méthodes de résolution

La complexité d'un problème d'ordonnancement d'une machine unique à vitesse variable, lorsque la machine n'a que deux états (ON et OFF), $(1, TOU|vitesses, q_j|TEC)$ est déjà étudiée dans [1]. Les auteurs ont prouvé que le problème est NP-difficile. Ainsi, la version généralisée de ce problème, considérée dans ce chapitre, avec trois états principaux et deux états de transition, $(1, TOU|états, vitesses, q_j|TEC)$ est aussi NP-difficile.
Les méthodes exactes ne sont pas en mesure de résoudre les instance de grande taille. C'est pourquoi, nous avons proposé une heuristique, un algorithme génétique, ainsi qu'un algorithme mémétique pour les résoudre.

Le principe de l'heuristique proposée est basé sur l'allocation des états non processeurs à la machine pendant les périodes supplémentaires (la même idée que l'heuristique présentée dans la section précédente (section 3.7.1)).
Contrairement au problème examiné dans la section précédente, nous supposons des délais de traitement différents pour chacune des tâches. Par conséquent, pour obtenir le nombre maximum de périodes supplémentaires calculé par l'équation 5.1, le nombre minimum requis de périodes pour les temps de traitement doit être remplacé par P. Pour cela, le temps de traitement de chaque travail avec la vitesse maximale (p_{j,v_j}) est sélectionné. Ainsi, en considérant les paramètres supposés, la valeur de x peut être obtenue comme suit :

$$x = T - (\beta_1 + \beta_2 + 1) - \sum_{j=1}^{n} p_{j,v_j} \tag{5.11}$$

Par exemple, dans un problème avec 3 tâches, 30 périodes et 3 vitesses, avec les valeurs des paramètres comme suit : $p_1 = \{6, 4, 2\}, p_2 = \{5, 4, 3\}, p_3 = \{5, 3, 1\}, \beta_1 = 2, \beta_2 = 1$, la valeur x est égale à [30-(2+1+1)-(2+3+1)]=20.

Comme nous avons utilisé un algorithme génétique pour évaluer la performance de notre heuristique pour le problème avec la même consommation énergétique des taches, nous avons également développé un algorithme génétique pour le problème étudié dans ce chapitre.
Toute solution à ce problème, est un planning sur un horizon temporel allant de la période 0 à la

1	1	1	1	2	2	110	110	110	230	230	320	320	320	320	4	1	1	1	2	2	430	430	520	520	520	4	1	1	1	1	1	1

FIGURE 5.8 – Le chromosome de notre algorithme génétique

période T, qui définit l'état de la machine pendant chaque période. Ainsi, dans cet article, chaque chromosome de l'algorithme génétique est représenté par des gènes $T + 1$ et chaque gène identifie l'état de la machine dans une période. Pour distinguer les états de la machine, chaque état est représenté par un nombre spécifique comme $OFF = 1$, $Ton = 2$, $Idle = 3$, et $Toff = 4$. De plus, un nombre entier supérieur à 100 ($k > 100$) représente que la machine est en mode Etat ON. En d'autres termes, si dans la période t, la machine traite le job j avec speed i, dans le chromosome correspondant, le gène t se remplit avec le nombre $(100 * j + 10 * i)$. Figure 5.8 représente le chromosome correspondant de l'instance présentée à Figure 4.2. Puisque dans ce cas le nombre de périodes est de 32, ce chromosome est donc constitué de 33 gènes. Le nombre 230 dans le gène 10 signifie qu'au cours de la période 9, la machine traite le travail 2 avec la vitesse 3 et le nombre 4 dans le gène 10 avec la vitesse 3. 27ème gène signifie que pendant la période 26 la machine est en état Toff.

La performance de l'algorithme génétique dépend de la sélection de ses principaux paramètres (la taille de la population, le taux de croisement, le taux de mutation, un opérateur de croisement, un opérateur de mutation et le nombre d'itérations) en fonction du problème étudié. Dans cette étude, l'opérateur de sélection de la roulette a été choisi comme opérateur de sélection des parents pour produire les nouveaux chromosomes. Dans cette thèse, un plan d'experience de Taguchi est utilisée au lieu d'un plan expérimental factoriel complet pour déterminer les paramètres de l'algorithme génétique. A cette fin, six facteurs sont considérés comme tels, l'un d'eux ayant deux niveaux et les autres trois niveaux (Tableau 5.1).

Les résultats de l'analyse de la méthode taguchi sont donnés dans la Figure 5.9. Comme résultat, un seul point est choisi comme méthode de croisement pour produire deux nouveaux enfants, et le taux de croisement choisi est égal à 0,7 (70 à %). Le taux de mutation choisi est égal à 0,1 (10 %), et la taille de la population initiale et le nombre d'itérations sont considérés comme étant respectivement égaux à 150 et 100.

Pour améliorer la qualité des solutions obtenues par l'algorithme génétique pour ce problème,

Factor	level1	Level2	Level3
Crossover operator	single-point	double-point	-
Mutation operator	swap	revers	insert
Crossover rate	0.7	0.8	0.9
Mutation rate	0.05	0.10	0.15
Population size	50	100	150
Iteration number	50	100	150

TABLEAU 5.1 – Design factors and their levels

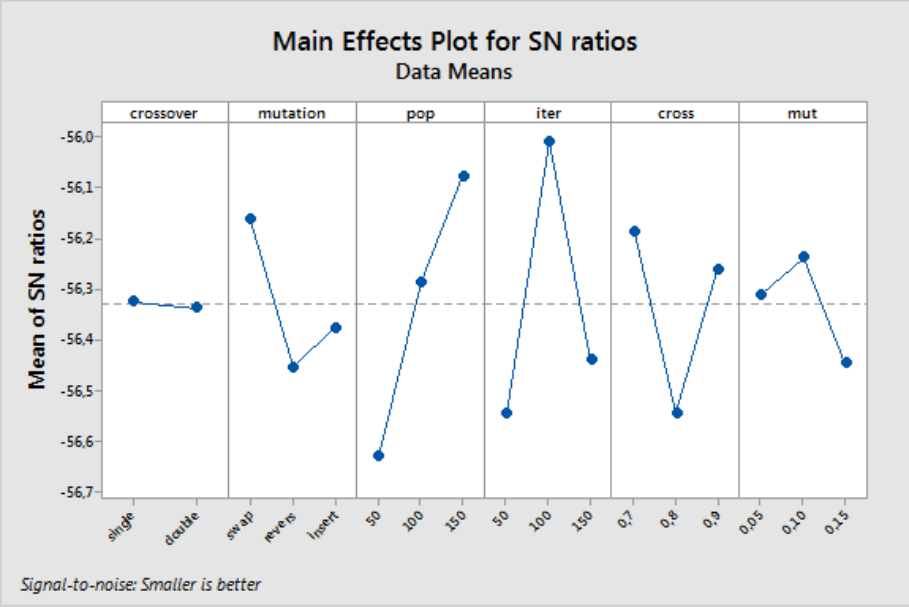

FIGURE 5.9 – Résultat de l'analyse Taguchi

une procédure de recherche locale est également introduite (algorithme memetic). La procédure de recherche locale est appliquée pour augmenter la qualité des 15 (10% de la taille de la population) meilleurs chromosomes de la population à chaque itération. Pour ce faire, une nouvelle solution sera créée en augmentant la vitesse de traitement d'une unité pour un travail et, par conséquent, en exécutant tous les autres travaux plus tôt que la différence entre les temps de traitement correspondants. Cette procédure doit être répétée pour toutes les tâches dans l'ordre de leur séquence. Ainsi, pour un problème avec n emplois, au plus n nouvelles solutions peuvent être créées à partir de chaque solution initiale. Enfin, la solution ayant la meilleure fonction objective doit remplacer la solution initiale. Par exemple, pour le chromosome considéré dans Figure 4.13, la première tâche était traitée à la première vitesse. Dans la première solution proposée par l'algorithme memetic, le

premier travail est traité à la deuxième vitesse, et ensuite, toutes les taches sont effectuées plus tôt.

Les performances des méthodes proposées ont été examinées par plusieurs instances numériques inspirées de la littérature ([61]). Pour cela, l'algorithme heuristique, l'algorithme génétique et l'algorithme mémétique ont été codés en langage C++ dans Visual Studio 2015, et le logiciel CPLEX est utilisé pour résoudre les instances avec la méthode exacte (Branch et Cut). Cinq exemples différents sont générés au hasard pour chaque taille d'instance en modifiant les temps de traitement et les consommations d'énergie des jobs, ainsi que l'unité du prix de l'énergie dans chaque période. Le temps de calcul avec CPLEX, pour toutes les expériences, a été fixé à 1 heure ou 3600 secondes. Pour les problèmes inférieurs à 15 travaux, 5 vitesses et 120 périodes, CPLEX a su trouver les solutions optimales.
En général, pour les problèmes de petite taille, l'heuristique trouve les solutions avec un écart de 17,1% en moyenne, alors que cet écart pour l'algorithme génétique et l'algorithme mémétique est égal à 7,5% et 2,7%, respectivement. Le temps de calcul moyen des problèmes de petite taille pour les algorithmes heuristiques, génétiques et mémétiques est égal à 1,06 s, 15,64 s et 18,50 s, respectivement.

Pour les problèmes de plus de 20 tâches, 5 vitesses et 160 périodes, CPLEX n'a pas été en mesure de trouver la solution optimale. Nous avons donc simplement comparé les solutions obtenues par ces trois algorithmes. Il faut mentionner que, dans tous les cas, l'algorithme mimétique trouve la meilleure solution. Les résultats montrent que l'écart moyen entre les solutions des algorithmes heuristiques et génétiques et la solution obtenue par MA, est d'environ 21,1% pour l'heuristique, et 18,4% pour le génétique.
De plus, une analyse de la variance (ANOVA) avec un niveau de confiance de 95 % a été réalisée à l'aide du logiciel Minitab.17 pour vérifier la validité statistique des résultats (figures 5.10 et 5.11).

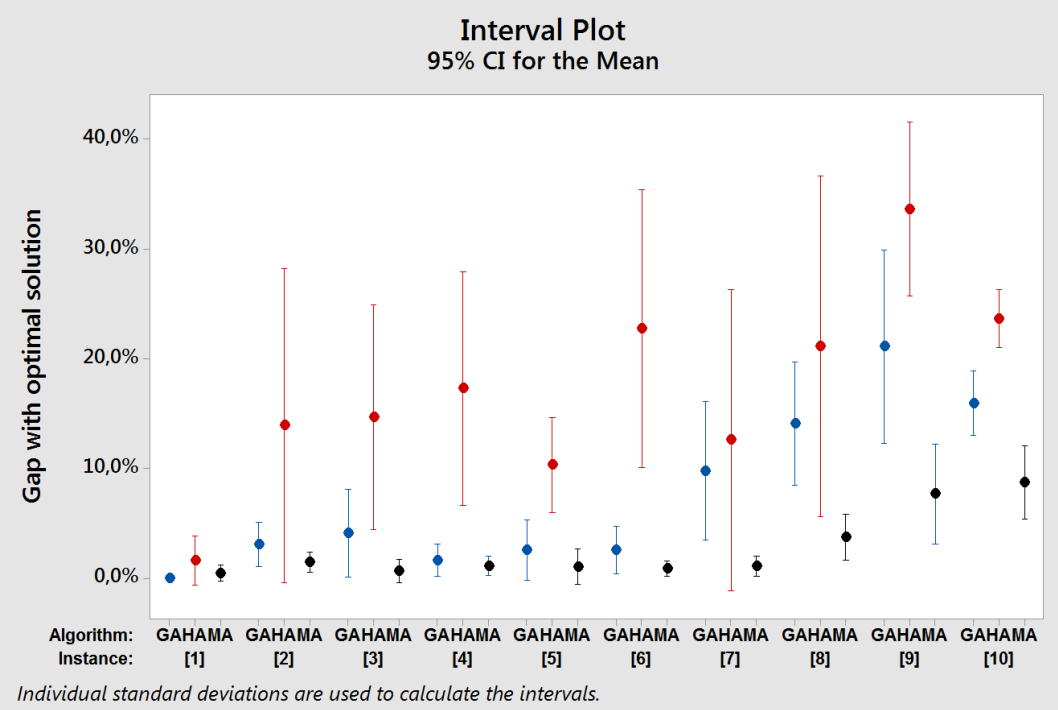

FIGURE 5.10 – Comparaison des performances de GA, HA et MA avec la solution optimale

Analyse de complexité des sous-problèmes

Comme il est présenté dans les sections précédentes, le problème d'ordonnancement de plusieurs taches sur une machine unique à plusieurs états et à plusieurs vitesses lorsque les travaux consomment différentes quantités d'énergie est NP-hard. Dans cette section, nous nous intéressons à l'analyse de la complexité de deux sous-versions de ce problème. Pour ce faire, nous avons considéré le cas de vitesse uniforme et le cas de vitesse du problème lorsque les séquences des travaux sont fixes $(1, TOU|états, séquence, q_j|TEC$ et $1, TOU|états, vitesses = v, séquence, q_j|TEC)$.

Pour analyser la complexité de ces problèmes, la même approche que celle que nous avons utilisée pour les problèmes $(1, TOU|états, séquence|TEC)$ et $(1, TOU|états, pmtn|TEC)$, est réalisée. Deux graphes finis différents sont donc proposés pour modéliser ces problèmes.

Pour le problème $(1, TOU|états, séquence, q_j|TEC)$, le nombre total de noeuds et d'arcs pour une instance avec des périodes T et des tâches n sont :

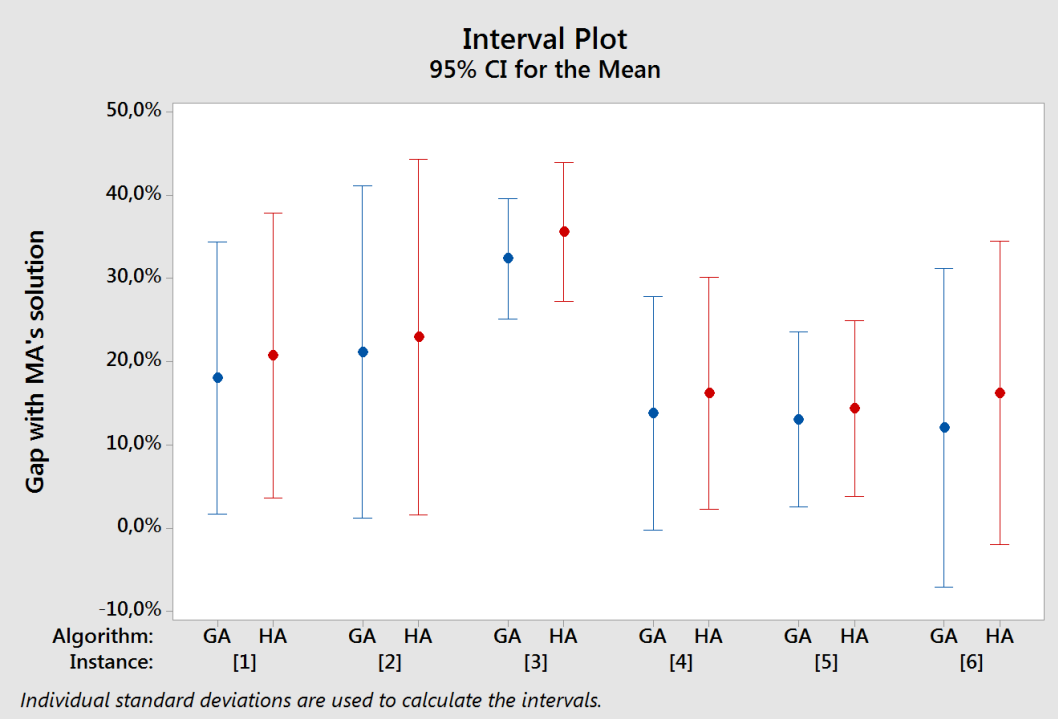

FIGURE 5.11 – L'amélioration des résultats par MA contre GA et HA pour les instances de grande taille

$$|V| = (n+2) * (x+1) \cong nT \tag{5.12}$$

$$|E| \cong T^2 n \tag{5.13}$$

Selon [77], la complexité de l'approche développée est égale à :

$$\begin{aligned} &O(T^2 n + Tn \log Tn) = O(T^2 n + Tn \log T + Tn \log n) \\ &\cong O(T^2 n) \end{aligned} \tag{5.14}$$

Car $n < T$ (le pire des cas pour tout problème réalisable), nous avons $\log n < \log T < T$. On peut donc conclure que le problème est un polynôme de degré 3 ou un problème polynomial cubique ($O(T^3)$).

Le deuxième sous-problème, est une version séquentielle fixe du problème général lorsque le nombre de vitesses est fixe pour tous les travaux $(1, TOU|états, vitesses = v, séquence, q_j|TEC)$. Dans ce cas, pour chaque tâche, nous devons choisir sa vitesse d'exécution parmi un ensemble de valeurs donné. Chaque vitesse correspond à une consommation d'énergie et à un temps de traitement donnés. Par conséquent, il pourrait être intéressant de traiter les travaux plus rapidement lorsque le coût énergétique est faible et de les traiter plus lentement lorsque le coût énergétique est élevé. Notez que pour satisfaire la non-prévention dans ce cas, la solution doit être composée d'une vitesse unique i pour chaque tâche j pour la traiter avec le temps de traitement associé p_j^i et la consommation électrique q_j^i non préemptivement.

Comme cas de vitesse uniforme de ce problème, un graphe avec les niveaux de décision $T+1$ est représenté pour modéliser le cas de vitesse scalable. Chaque niveau de décision (l) a un ensemble de noeuds (H_l) qui représente les derniers jobs possibles dans la séquence donnée J_j, qui est traité avec la vitesse i jusqu'au moment l $(J_j^i \quad ; \forall j \in \{1, \cdots, n\}, i \in \{1, \cdots, v)\}$. Par conséquent, en considérant les états initial-OFF et final-OFF, le graphe est composé de $(n * v) + 2$ différents types de nœuds, où n représente le nombre de tâches, et v représente le nombre de vitesses. Le nombre de périodes sans traitement pour ce cas est obtenu par la formulation suivante :

$$x' = T - \sum_{j=1}^{n} p_j^v - (\beta_1 + \beta_2 + 1) \tag{5.15}$$

Dans tout le graphe, il y a au plus $(x' + 1)$ des noeuds avec le même numéro de noeud $(\{I, J_1^i, J_2^i, \cdots, J_n^i, F\})$. Ainsi, le nombre total de nœuds pour un problème avec des périodes T, des tâches n et des vitesses v est :

$$|V| \leq ((n * v) + 2) * (x' + 1) \cong n \cdot v \cdot T \tag{5.16}$$

Le nombre total d'arcs pour un problème de vitesse évolutif avec des périodes T, des tâches n et des vitesses v est :

$$|E| \cong T^2 n v^2 \tag{5.17}$$

La complexité de l'algorithme de Dijkstra pour le problème de l'échelle de vitesse peut être calculée par la formulation suivante :

$$O(T^2nv^2 + Tnv \log Tnv) \tag{5.18}$$

Car $n < T$, nous avons $\log n < \log T < \log T < T$, $\log v < v$, et il n'y a aucune limitation pour v. On peut donc conclure que le cas de ce problème à l'échelle de la vitesse est tout au plus pseudo polynomial.

Conclusions et perspectives

Cette thèse présente des approches de modélisation et de résolution des problèmes de planification de plusieurs tâches sur une seule machine multi-états dans le but de minimiser les coûts de consommation d'énergie. Pour ce faire, une étude complète des différents problèmes d'ordonnancement des concepts énergétiques avec aspects énergétiques est réalisée dans la section II.

Comme première contribution de cette thèse, nous avons proposé un modèle mathématique amélioré pour un problème existant dans la littérature ([61]), et nous avons analysé la complexité du problème en utilisant une nouvelle approche de programmation dynamique. Nous avons prouvé que contrairement à ce qui est considéré précédemment dans [61], le problème d'ordonnancement de plusieurs tâches avec une séquence prédéterminée sur une seule machine avec multi-états $(1, TOU|états,\ séquence|TEC)$ est polynomial.

Dans la section II, nous avons étudié une version générale du problème précédent pour optimiser l'ordonnancement des tâches, ainsi que l'état de la machine à chaque période (problème $1, TOU|états|TEC$). Après avoir montré que le problème est NP-difficile, nous avons présenté un modèle mathématique pour ce problème. Ensuite, nous avons proposé une heuristique et un algorithme génétique résoudre les problèmes de grande taille. Nous avons également défini quatre bornes inférieures pour ce problème. De plus, la complexité de quatre sous-problèmes est également étudiée.

Dans la section II, nous avons étudié le problème d'ordonnancement plus complexe avec différentes consommations d'énergie des tâches et des vitesses variables de la machine $(1, TOU|états,\ vitesses,\ q_j|TEC)$. Nous avons proposé deux modèles mathématiques pour ce problème et trois algorithmes (une heuristique, un génétique et un mémétique) pour résoudre les instances de moyenne et grande

taille.

Cette thèse porte sur la modélisation et l'optimisation de trois types de problèmes d'ordonnancement sur seule machine avec plusieurs états afin de minimiser les coûts totaux liés à consommation d'énergie. Dans ce qui suit, nous discutons des limites de ces travaux de recherche et suggérons d'autres perspectives et pistes de recherche comme future extension possible.

Pour les perspectives à court terme, nous sommes intéressés à améliorer la performance des algorithmes présentés pour le problème $1, TOU|états,\ vitesses,\ q_j|TEC$ et développer d'autres méthodes exactes pour ce problème. Par exemple, il pourrait être intéressant de fournir quelques règles de dominance pour améliorer la performance de la méthode de la branche et de la méthode liée pour ce problème. De plus, nous cherchons à proposer une autre méthode de recherche locale pour améliorer les performances de l'algorithme memetic proposé. De plus, nous voulons généraliser le modèle pour tenir compte des dates de publication et des dates d'échéance pour chaque emploi. La prise en compte des problèmes multi-objectifs, qui optimisent les coûts totaux de consommation d'énergie en plus d'un objectif de planification traditionnel comme makespan semble pertinente et prometteuse.

Bibliography

[1] K. Fang et al. “Scheduling on a single machine under time-of-use electricity tariffs”. In: *Annals of Operations Research* (2014), pp. 1–29 (cit. on pp. 3, 27, 109, 142, 160).

[2] L. Merkert et al. “Scheduling and energy – Industrial challenges and opportunities”. In: *Computers & Chemical Engineering* 72 (2015). A Tribute to Ignacio E. Grossmann, pp. 183–198 (cit. on pp. 6, 17, 144).

[3] C. Gahm et al. “Energy-efficient scheduling in manufacturing companies: A review and research framework”. In: *European Journal of Operational Research* 248.3 (2016), pp. 744–757. ISSN: 0377-2217 (cit. on pp. 6, 17, 144).

[4] S. Willeke, G. Ullmann, and P. Nyhuis. “Method for an Energy-Cost-Oriented Manufacturing Control to Reduce Energy Costs: Energy Cost Reduction by Using a New Sequencing Method”. In: *2016 International Conference on Industrial Engineering, Management Science and Application (ICIMSA)*. IEEE. 2016, pp. 1–5 (cit. on p. 10).

[5] M. Rodoplu, T. Arbaoui, and A. Yalaoui. “Energy Contract Optimization for the Single Item Lot Sizing Problem in a Flow-Shop Configuration and Multiple Energy Sources”. In: *IFAC-PapersOnLine* 51.11 (2018), pp. 1089–1094 (cit. on p. 10).

[6] O. Masmoudi et al. “Lot-sizing in a multi-stage flow line production system with energy consideration”. In: *International Journal of Production Research* 55.6 (2017), pp. 1640–1663 (cit. on p. 10).

[7] O. Masmoudi et al. “Solving a capacitated flow-shop problem with minimizing total energy costs”. In: *The International Journal of Advanced Manufacturing Technology* 90.9-12 (2017), pp. 2655–2667 (cit. on p. 10).

[8] H. Luo et al. “Hybrid flow shop scheduling considering machine electricity consumption cost”. In: *International Journal of Production Economics* 146.2 (2013), pp. 423–439 (cit. on pp. 11, 26).

[9] H. Y. Fuchigami and S. Rangel. “A survey of case studies in production scheduling: Analysis and perspectives”. In: *Journal of Computational Science* 25 (2018), pp. 425–436 (cit. on p. 12).

[10] J. Y. Leung. *Handbook of scheduling: algorithms, models, and performance analysis*. CRC Press, 2004 (cit. on pp. 12, 15, 144).

[11] P. Brucker and P. Brucker. *Scheduling algorithms*. Vol. 3. Springer, 2007 (cit. on pp. 13, 15).

[12] M. L. Pinedo. *Scheduling: theory, algorithms, and systems*. Springer, 2016 (cit. on pp. 13, 14, 16).

[13] R. L. Graham et al. “Optimization and approximation in deterministic sequencing and scheduling: a survey”. In: *Annals of discrete mathematics*. Vol. 5. Elsevier, 1979, pp. 287–326 (cit. on pp. 14, 15, 144).

[14] M. R. Garey. “DS Johnson Computers and intractability”. In: *A Guide to the Theory of NP-Completeness* (1979) (cit. on pp. 16, 66).

[15] E. L. Lawler et al. “Sequencing and scheduling: Algorithms and complexity”. In: *Handbooks in operations research and management science* 4 (1993), pp. 445–522 (cit. on p. 16).

[16] K. Biel and C. H. Glock. “Systematic literature review of decision support models for energy-efficient production planning”. In: *Computers & Industrial Engineering* 101 (2016), pp. 243–259 (cit. on pp. 17, 144).

[17] X. Wang et al. “A low-carbon production scheduling system considering renewable energy”. In: *Service Operations, Logistics, and Informatics (SOLI), 2011 IEEE International Conference on*. IEEE. 2011, pp. 101–106 (cit. on p. 18).

[18] C. Liu et al. “Sustainable performance oriented operational decision-making of single machine systems with deterministic product arrival time”. In: *Journal of Cleaner Production* 85 (2014), pp. 318–330 (cit. on p. 18).

[19] C.-H. Liu and D.-H. Huang. “Reduction of power consumption and carbon footprints by applying multi-objective optimisation via genetic algorithms”. In: *International Journal of Production Research* 52.2 (2014), pp. 337–352 (cit. on pp. 18, 20).

[20] C.-H. Liu. “Mathematical programming formulations for single-machine scheduling problems while considering renewable energy uncertainty”. In: *International Journal of Production Research* 54.4 (2016), pp. 1122–1133 (cit. on p. 18).

[21] C.-H. Liu. “Approximate trade-off between minimisation of total weighted tardiness and minimisation of carbon dioxide (CO2) emissions in bi-criteria batch scheduling problem”. In: *International Journal of Computer Integrated Manufacturing* 27.8 (2014), pp. 759–771 (cit. on p. 18).

[22] K. Fang et al. “A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction”. In: *Journal of Manufacturing Systems* 30.4 (2011), pp. 234–240 (cit. on p. 18).

[23] D. Lei and X. Guo. “An effective neighborhood search for scheduling in dual-resource constrained interval job shop with environmental objective”. In: *International Journal of Production Economics* 159 (2015), pp. 296–303 (cit. on p. 18).

[24] M. Mezmaz et al. “A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems”. In: *Journal of Parallel and Distributed Computing* 71.11 (2011), pp. 1497–1508 (cit. on p. 18).

[25] G. Mouzon, M. B. Yildirim, and J. Twomey. “Operational methods for minimization of energy consumption of manufacturing equipment”. In: *International Journal of Production Research* 45.18-19 (2007), pp. 4247–4271 (cit. on pp. 19, 20).

[26] Y. He et al. “A modeling method of task-oriented energy consumption for machining manufacturing system”. In: *Journal of Cleaner Production* 23.1 (2012), pp. 167–174 (cit. on p. 19).

[27] M. Ji, J.-Y. Wang, and W.-C. Lee. “Minimizing resource consumption on uniform parallel machines with a bound on makespan”. In: *Computers & Operations Research* 40.12 (2013), pp. 2970–2974 (cit. on p. 19).

[28] Z. Jiang, L. Zuo, and E. Mingcheng. “Study on multi-objective flexible job-shop scheduling problem considering energy consumption”. In: *Journal of Industrial Engineering and Management* 7.3 (2014), p. 589 (cit. on p. 19).

[29] Y. Liu et al. “An investigation into minimising total energy consumption and total weighted tardiness in job shops”. In: *Journal of Cleaner Production* 65 (2014), pp. 87–96 (cit. on p. 19).

[30] C. Lu et al. “Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm”. In: *Journal of cleaner production* 144 (2017), pp. 228–238 (cit. on p. 19).

[31] S. A. Mansouri, E. Aktas, and U. Besikci. “Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption”. In: *European Journal of Operational Research* 248.3 (2016), pp. 772–788 (cit. on p. 20).

[32] S. Afshin Mansouri and E. Aktas. “Minimizing energy consumption and makespan in a two-machine flowshop scheduling problem”. In: *Journal of the Operational Research Society* 67.11 (2016), pp. 1382–1394 (cit. on p. 20).

[33] Y. Liu et al. “A multi-objective genetic algorithm for optimisation of energy consumption and shop floor production performance”. In: *International Journal of Production Economics* 179 (2016), pp. 259–272 (cit. on p. 20).

[34] M. B. Yildirim and G. Mouzon. “Single-machine sustainable production planning to minimize total energy consumption and total completion time using a multiple objective genetic algorithm”. In: *Engineering Management, IEEE Transactions on* 59.4 (2012), pp. 585–597 (cit. on p. 20).

[35] G. Mouzon and M. B. Yildirim. “A framework to minimise total energy consumption and total tardiness on a single machine”. In: *International Journal of Sustainable Engineering* 1.2 (2008), pp. 105–116 (cit. on p. 20).

[36] A. Che et al. “Energy-efficient bi-objective single-machine scheduling with power-down mechanism”. In: *Computers & Operations Research* 85 (2017), pp. 172–183 (cit. on p. 20).

[37] G. Chen et al. “Energy-efficient production systems through schedule-based operations”. In: *IEEE Transactions on Automation Science and Engineering* 10.1 (2013), pp. 27–37 (cit. on p. 20).

[38] M. Dai et al. “Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm”. In: *Robotics and Computer-Integrated Manufacturing* 29.5 (2013), pp. 418–429 (cit. on p. 21).

[39] Y. He et al. “An energy-responsive optimization method for machine tool selection and operation sequence in flexible machining job shops”. In: *Journal of Cleaner Production* 87 (2015), pp. 245–254 (cit. on p. 21).

[40] P. Liang et al. “An ant optimization model for unrelated parallel machine scheduling with energy consumption and total tardiness”. In: *Mathematical Problems in Engineering* 2015 (2015) (cit. on p. 21).

[41] X. Liu, F. Zou, and X. Zhang. “Mathematical model and genetic optimization for hybrid flow shop scheduling problem based on energy consumption”. In: *Control and Decision Conference, 2008. CCDC 2008. Chinese*. IEEE. 2008, pp. 1002–1007 (cit. on p. 21).

[42] M. Nattaf, C. Artigues, and P. Lopez. “A hybrid exact method for a scheduling problem with a continuous resource and energy constraints”. In: *Constraints* 20.3 (2015), pp. 304–324 (cit. on p. 21).

[43] M. Nattaf et al. “Energetic reasoning and mixed-integer linear programming for scheduling with a continuous resource and linear efficiency functions”. In: *OR spectrum* 38.2 (2016), pp. 459–492 (cit. on p. 21).

[44] A. Antoniadis and C.-C. Huang. “Non-preemptive speed scaling”. In: *Journal of Scheduling* 16.4 (2013), pp. 385–394 (cit. on p. 21).

[45] A. Antoniadis, C.-C. Huang, and S. Ott. “A fully polynomial-time approximation scheme for speed scaling with sleep state”. In: *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2015, pp. 1102–1113 (cit. on p. 21).

[46] E. Bampis et al. “From preemptive to non-preemptive speed-scaling scheduling”. In: *Discrete Applied Mathematics* 181 (2015), pp. 11–20 (cit. on p. 21).

[47] J. Cohen, D. Cordeiro, and P. L. F. Raphael. “Energy-aware multi-organization scheduling problem”. In: *European Conference on Parallel Processing*. Springer. 2014, pp. 186–197 (cit. on p. 21).

[48] G.-S. Liu, H.-D. Yang, and M.-B. Cheng. “A three-stage decomposition approach for energy-aware scheduling with processing-time-dependent product quality”. In: *International Journal of Production Research* 55.11 (2017), pp. 3073–3091 (cit. on p. 22).

[49] A. Faruqui, R. Hledik, and J. Palmer. *Time-varying and dynamic rate design*. Regulatory Assistance Project, 2012 (cit. on p. 22).

[50] X. Gong et al. “An energy-cost-aware scheduling methodology for sustainable manufacturing”. In: *Procedia CIRP* 29 (2015), pp. 185–190 (cit. on pp. 24–27).

[51] J.-Y. Ding et al. “Parallel machine scheduling under time-of-use electricity prices: New models and optimization approaches”. In: *IEEE Transactions on Automation Science and Engineering* 13.2 (2016), pp. 1138–1154 (cit. on p. 24).

[52] M. Tan et al. “Optimal hot rolling production scheduling for economic load dispatch under time-of-use electricity pricing”. In: *arXiv preprint arXiv:1501.05502* (2015) (cit. on p. 24).

[53] Y. Tan and S. Liu. “Models and optimisation approaches for scheduling steelmaking–refining–continuous casting production under variable electricity price”. In: *International Journal of Production Research* 52.4 (2014), pp. 1032–1049 (cit. on p. 24).

[54] P. M. Castro, I. Harjunkoski, and I. E. Grossmann. “New continuous-time scheduling formulation for continuous plants under variable electricity cost”. In: *Industrial & engineering chemistry research* 48.14 (2009), pp. 6701–6714 (cit. on p. 25).

[55] M. Burcea et al. “Scheduling for electricity cost in smart grid”. In: *Combinatorial Optimization and Applications*. Springer, 2013, pp. 306–317 (cit. on p. 25).

[56] H. Zhang, F. Zhao, and J. W. Sutherland. “Scheduling of a Single Flow Shop for Minimal Energy Cost Under Real-Time Electricity Pricing”. In: *Journal of Manufacturing Science and Engineering* 139.1 (2017), p. 014502 (cit. on p. 26).

[57] X. Gong et al. “Integrating labor awareness to energy-efficient production scheduling under real-time electricity pricing: An empirical study”. In: *Journal of Cleaner Production* 168 (2017), pp. 239–253 (cit. on pp. 26, 34, 145).

[58] J.-Y. Moon, K. Shin, and J. Park. “Optimization of production scheduling with time-dependent and machine-dependent electricity cost for industrial energy efficiency”. In: *The International Journal of Advanced Manufacturing Technology* 68.1-4 (2013), pp. 523–535 (cit. on p. 26).

[59] J.-Y. Moon and J. Park. “Smart production scheduling with time-dependent and machine-dependent electricity cost by considering distributed energy resources and energy storage”. In: *International Journal of Production Research* 52.13 (2014), pp. 3922–3939 (cit. on p. 26).

[60] S. Albers and H. Fujiwara. "Energy-efficient algorithms for flow time minimization". In: *ACM Transactions on Algorithms (TALG)* 3.4 (2007), p. 49 (cit. on p. 27).

[61] F. Shrouf et al. "Optimizing the production scheduling of a single machine to minimize total energy consumption costs". In: *Journal of Cleaner Production* 67 (2014), pp. 197–207 (cit. on pp. 27, 36, 40, 47, 56–59, 62, 83, 91, 94, 108, 129, 135, 145, 147, 152, 156, 163, 167).

[62] G. Wan and X. Qi. "Scheduling with variable time slot costs". In: *Naval Research Logistics (NRL)* 57.2 (2010), pp. 159–171 (cit. on p. 27).

[63] I. Koutsopoulos and L. Tassiulas. "Control and optimization meet the smart power grid: Scheduling of power demands for optimal energy management". In: *Proceedings of the 2nd International Conference on Energy-Efficient Computing and Networking*. ACM. 2011, pp. 41–50 (cit. on p. 27).

[64] A. Che, Y. Zeng, and K. Lyu. "An efficient greedy insertion heuristic for energy-conscious single machine scheduling problem under time-of-use electricity tariffs". In: *Journal of Cleaner Production* 129 (2016), pp. 565–577 (cit. on p. 27).

[65] K. Li et al. "Parallel machine scheduling problems in green manufacturing industry". In: *Journal of Manufacturing Systems* 38 (2016), pp. 98–106 (cit. on p. 27).

[66] X. Gong et al. "A generic method for energy-efficient and energy-cost-effective production at the unit process level". In: *Journal of Cleaner Production* 113 (2016), pp. 508–522 (cit. on p. 27).

[67] Y. Mikhaylidi, H. Naseraldin, and L. Yedidsion. "Operations scheduling under electricity time-varying prices". In: *International Journal of Production Research* 53.23 (2015), pp. 7136–7157 (cit. on p. 27).

[68] X. Gong et al. "A power data driven energy-cost-aware production scheduling method for sustainable manufacturing at the unit process level". In: *Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on*. IEEE. 2016, pp. 1–8 (cit. on p. 27).

[69] S. Wang et al. "Bi-objective optimization of a single machine batch scheduling problem with energy cost consideration". In: *Journal of Cleaner Production* 137 (2016), pp. 1205–1215 (cit. on p. 27).

[70] C. Artigues, P. Lopez, and A. Hait. "The energy scheduling problem: Industrial case-study and constraint propagation techniques". In: *International Journal of Production Economics* 143.1 (2013), pp. 13–23 (cit. on p. 28).

[71] C. Artigues and P. Lopez. "Energetic reasoning for energy-constrained scheduling with a continuous resource". In: *Journal of Scheduling* 18.3 (2015), pp. 225–241 (cit. on p. 28).

[72] P. M. Castro, I. Harjunkoski, and I. E. Grossmann. "Optimal scheduling of continuous plants with energy constraints". In: *Computers & chemical engineering* 35.2 (2011), pp. 372–387 (cit. on p. 28).

[73] A. Bruzzone et al. "Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops". In: *CIRP Annals-Manufacturing Technology* 61.1 (2012), pp. 459–462 (cit. on p. 28).

[74] M. H. Dupty, P. Agrawal, and S. Rao. "Scheduling Under Power and Energy Constraints". In: *arXiv preprint arXiv:1609.07354* (2016) (cit. on p. 28).

[75] K. Fang et al. "Flow shop scheduling with peak power consumption constraints". In: *Annals of Operations Research* 206.1 (2013), pp. 115–145 (cit. on p. 28).

[76] K. Fang et al. “Scheduling on a single machine under time-of-use electricity tariffs”. In: *Annals of Operations Research* 238.1-2 (2016), pp. 199–227 (cit. on pp. 34, 145).

[77] M. L. Fredman and R. E. Tarjan. “Fibonacci heaps and their uses in improved network optimization algorithms”. In: *Journal of the ACM (JACM)* 34.3 (1987), pp. 596–615 (cit. on pp. 55, 114, 117, 151, 165).

[78] D. E. Goldberg. “Genetic algorithms in search, optimization, and machine learning, 1989”. In: *Reading: Addison-Wesley* (1989) (cit. on p. 90).

[79] V. Kodaganallur, A. K. Sen, and S. Mitra. “Application of graph search and genetic algorithms for the single machine scheduling problem with sequence-dependent setup times and quadratic penalty function of completion times”. In: *Computers & Industrial Engineering* 67 (2014), pp. 10–19 (cit. on p. 90).

[80] N.-Q. Nguyen et al. “Solving a malleable jobs scheduling problem to minimize total weighted completion times by mixed integer linear programming models”. In: *Asian Conference on Intelligent Information and Database Systems*. Springer. 2016, pp. 286–295 (cit. on p. 107).

[81] P. Garg. “A Comparison between Memetic algorithm and Genetic algorithm for the cryptanalysis of Simplified Data Encryption Standard algorithm”. In: *arXiv preprint arXiv:1004.0574* (2010) (cit. on p. 128).

List of publications

Outline of the current chapter

The following journal and conference papers have been produced as parts of outcomes of this research:

Refereed publications

- Aghelinejad, M., Ouazene, Y., & Yalaoui, A. (2017). Production scheduling optimisation with machine state and time-dependent energy costs. International Journal of Production Research, 1-18.
- Aghelinejad, M., Ouazene, Y., & Yalaoui, A. Production scheduling optimization of a single machine to minimize total energy consumption costs. International Journal of Theoretical Computer Science, submitted 2018
- Aghelinejad, M., Ouazene, Y., & Yalaoui, A. Complexity analysis of energy-efficient single machine scheduling problems. International Journal of Operations Research Perspectives, submitted 2018
- Aghelinejad, M., Ouazene, Y., & Yalaoui, A. Energy-efficient scheduling of a speed scalable and multi-states single machine. In process

Conference publications

- Aghelinejad, M., Ouazene, Y., & Yalaoui, A. (2016, December). Machine and production scheduling under electricity time varying prices. In Industrial Engineering and Engineering Management (IEEM), 2016 IEEE International Conference on (pp. 992-996). IEEE
- Aghelinejad, M. M., Ouazene, Y., & Yalaoui, A. (2017, September). Preemptive Scheduling of a Single Machine with Finite States to Minimize Energy Costs. In International Conference on Optimization and Decision Science (pp. 591-599). Springer, Cham.
- Aghelinejad, M. M, Ouazene, Y., & Yalaoui, A. (2018, June). Energy efficient scheduling problems under Time-Of-Use tariffs with different energy consumption of the jobs. IFAC-PapersOnLine, 51(11), 1053-1058.

- Aghelinejad, M. M., Ouazene, Y., & Yalaoui, A. (2018, September). Energy optimization of a speed-scalable and multi-states single machine scheduling problem. In International Conference on Optimization and Decision Science (ODS 2018)
- Aghelinejad, M. M., Ouazene, Y., & Yalaoui, A. (2018, October). Energy efficient scheduling of a multi-states and multi-speeds single machine system. In International conference on Metaheuristics and Nature Inspired Computing,(META 2018)

Mohammadmohsen AGHELINEJAD

Doctorat : Optimisation et Sûreté des Systèmes

Année 2018

Problèmes de planification et d'ordonnancement de la production sous les aspects énergétiques

Ces dernières années, les évolutions économiques et sociétales ont entraîné une augmentation rapide de la consommation d'énergie. De plus, dans presque tous les pays industriels, le prix de l'électricité a augmenté continuellement. L'amélioration de l'efficacité énergétique et la réduction de la consommation d'électricité jouent donc un rôle très important dans les industries modernes. Récemment, plusieurs études ont été consacrées à l'intégration de l'efficacité énergétique dans les problèmes d'ordonnancement. Cependant, seuls certains travaux précédents considéraient un problème d'une seule machine multi-états avec différents prix d'énergie. Cette thèse aborde le cas d'un problème d'ordonnancement d'une seule machine avec différents états de fonctionnement caractérisés par différentes vitesses et consommations d'énergie. Cette thèse vise à fournir une étude complète pour étudier la complexité et les méthodes d'optimisation des différentes variantes de ce problème. Dans ce but, une nouvelle approche de programmation dynamique basée sur un graphe fini est utilisée pour modéliser plusieurs problèmes et établir leur complexité. Nous abordons également différentes formulations mathématiques et méthodes d'optimisation pour résoudre les variantes NP-difficile de ce problème. Des méthodes approchées sont proposées pour résoudre le problème d'ordonnancement d'une seule machine avec multi-états et les coûts d'électricité de temps d'utilisation dans le cas avec vitesse unique ainsi que dans le cas avec vitesse multiple et différentes consommations d'énergie pour les tâches.

Mots clés : optimisation mathématique – ordonnancement (gestion) – consommation d'énergie – métaheuristiques.

Production Planning and Scheduling Problems under Energy Aspects

In the last few years, economic and societal developments have led to a rapid increase in energy consumption. Moreover, in almost every industrial nation, electricity prices, one of the main energy sources used in the manufacturing factories, have been rising continuously. Therefore, improving electric power efficiency and saving electricity plays a very important role in modern industries. Recently, several studies are devoted to integrate energy efficiency into the scheduling problems. However, only few of them considered a multi-states machine-scheduling problem under time varied energy prices. Therefore, this thesis deals with the case of a single machine scheduling problem with different operating states characterized by different speeds and energy consumptions. It aims to provide a complete study to investigate the complexity and the optimization methods of different variant of this problem. For this purpose, a dynamic programming approach based on a finite graph is used to model several problems and establish their complexities. We also address different mathematical formulations and optimization methods to solve the NP-hard variants of this problem. A new heuristic algorithm and a genetic algorithm are proposed for the single machine scheduling problem with multi-states and the Time-Of-Use electricity (TOU) costs. A heuristic algorithm, a genetic algorithm as well as a memetic algorithm are proposed for the single machine scheduling problem with multi-states, multi-speeds, TOU costs and different energy consumptions for the jobs.

Keywords: mathematical optimization – production scheduling – energy consumption – metaheuristics.

Thèse réalisée en partenariat entre :

Région ALSACE CHAMPAGNE-ARDENNE LORRAINE

UNION EUROPÉENNE

l'Europe s'engage en Champagne-Ardenne avec le FEDER

utt UNIVERSITÉ DE TECHNOLOGIE TROYES

Ecole Doctorale "Sciences pour l'Ingénieur"