Titre: Title:	Optimal Operator Assignment in a Real-life Environment
Auteur: Author:	Nazanin Haghjoo
Date:	2023
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Haghjoo, N. (2023). Optimal Operator Assignment in a Real-life Environment [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/53439/

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Optimal operator assignment in a real-life environment

NAZANIN HAGHJOO

Département de mathématiques et de génie industriel

Mémoire présenté en vue de l'obtention du diplôme de Maîtrise ès sciences appliquées

Génie industriel

Mai 2023

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé:

Optimal operator assignment in a real-life environment

présenté par **Nazanin HAGHJOO**

en vue de l'obtention du diplôme de Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

Michel GAMACHE, président
Michel GENDREAU, membre et directeur de recherche
Antoine LEGRAIN, membre et codirecteur de recherche
Gilles PESANT, membre

DEDICATION

To all the women of my land, Iran,

who do not stop progressing

despite all the hardships and gender inequalities

ACKNOWLEDGEMENTS

First, I would like to show my appreciation to my supervisors Prof. Michel Gendreau and Prof. Antoine Legrain, for their helpful advice, professional supervision, and availability. I was fortunate to get the help of their knowledge and experience.

I also want to thank JITbase company for letting me carry out the project by providing a comprehensive and precise description of their requirement, sending me the required information, and answering my question patiently. I would like to thank Prof. Michel Gamache and Prof. Gilles Pesant, who agreed to be a part of my jury.

I am thankful to my parents, my sister, and all my friends for their support.

RÉSUMÉ

Le problème d'affectation d'un opérateur est un problème d’optimisation spécifique qui vise à trouver l’allocation optimale d’un opérateur à plusieurs tâches définies. Cette étude porte sur la proposition d'un modèle et d'une solution de programmation par contraintes (PPC) en réponse à un besoin de la société JITbase. Le problème est défini pour des machines à commande numériques par calculateur (CNC) parallèles non identiques, et l'objectif est d'affecter un seul opérateur aux machines CNC pour effectuer deux types de tâches: superviser les machines lorsque nécessaire et effectuer plusieurs opérations manuelles en parallèle.

La PPC est un outil pratique pour modéliser les problèmes d’ordonnancement. JITbase a été confronté au défi d'affecter un seul opérateur aux tâches régulières et aux opérations manuelles, et nous résolvons ce défi en fournissant un nouveau modèle de PPC. Notre partenaire industriel utilise déjà un outil basée sur la PPC pour son problème; par conséquent, nous améliorons leur modèle de PPC avec l’ajout de nouvelles caractristiques. Ce nouveau modèle élabore un plan pour affecter l'opérateur à la supervision des tâches et à l'exécution simultanée des tâches parallèles. La fonction objectif consiste minimiser le temps requis pour effectuer toutes les tâches sur toutes les machines. Nous testons également le modèle sur un ensemble de données réelles fournies par JITbase et présentons plusieurs analyses de sensibilité qui vont aider l'entreprise à amliorer le plus possible leurs opérations.

ABSTRACT

The operator assignment problem is a specific optimization problem that aims at finding the optimal allocation of the operator to several defined tasks. This study focuses on proposing a model and solution in response to a requirement for JITbase company. The problem is defined for non-identical parallel Computer-Numerical Control (CNC) machines, and the goal is to assign a single operator to the CNC machines to perform two jobs: supervising the machines when needed and conducting several manual operations in parallel.

Constraint Programming (CP) is a practical tool to model especially scheduling problems in a convenient manner. JITbase has been facing the challenge of assigning a single operator to both regular tasks and manual operations, and we resolve it by providing a new CP model. Our industrial partner already applies CP to model their problem without the parallel tasks; hence, we improve a CP model to modify it. Therefore, the new model considers a plan to assign the operator to supervise the tasks and perform parallel tasks simultaneously. The objective function is to minimize the makespan to ensure that the minimum delay, if any, is added to the scheduling plan due to the operator's tasks. We also tested the model on a JITbase real dataset and provide several sensitivity analyses that will be helpful for the company to improve its operation as much as possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

The industrial revolution saw a significant shift from manual labour to machine-based manufacturing. The machining industry has improved in such a manner that, in the third revolution, most machines require much less attention from humans than before. The dependency of machines on humans has been significantly limited by the introduction of Computer-Numerical Control (CNC) machines, which can perform automatic operations by using a computer program. The use of CNC machines provides benefits such as reducing lead time, removing operator errors, and decreasing labour costs (Pabla and Adithan, 1994).

Although automation has reduced the requirement for human resources, there is still a demand for the supervision of human resources, especially in manufacturing industries. MacCarthy et al. (2001) stated that despite the technological advancement of this decade, manufacturing enterprises still need human resources to contribute to production planning systems. They believed that manufacturing systems should have a framework to optimize the integration of human and computer-based production planning. Therefore, manufacturing industries require an optimal program for allocating human resources to CNC machines, with the aim of reducing human intervention and increasing operational efficiency. This study also focuses on finding this optimal human allocation to CNC machines.

Mathematical Optimization played a significant role in the Industrial Revolution. Optimization is a field of mathematics that tries to find the optimal solution to a given problem using a range of techniques. It was developed to improve efficiency and productivity in industries, enabling manufacturers to have quick and cost-effective production. One of the powerful tools for solving optimization problems is Constraint Programming (CP), which is practical in various domains, such as scheduling and allocation programs.

Despite the benefits of the automated industry in reducing the demand for human labour, companies still face a shortage of specialized operators to supervise CNC machines. Optimization models can help companies efficiently allocate operators to CNC machines to meet this demand. This can minimize the negative effects of the operator shortage.

In an automated manufacturing environment, a bar feeder is a device used to feed raw materials into a CNC machine. Despite the advantages of automation in reducing the need for human

intervention, an operator is still required to carry out certain manual operations, including reloading the bar feeder, conducting quality control checks, and performing tool changes. Hence, an optimized plan for the operator to supervise the tasks and conduct the manual operations requires a complex paradigm. This study aims to provide such a paradigm for an industrial partner named JITbase. JITbase is a company based in Montreal, QC, Canada that offers improvement software solutions to industries. The solutions allow the assignment of several CNC machine to an operator to manage the operator shortage and enhance the production margins. JITbase defined some regular duties and manual operations on each CNC machines to be performed by the operator.

According to the JITbase statement, all their information updates in real-time, allowing for immediate adjustments to short-term planning based on machine data. Whether an unexpected stop happens on a machine or a manual operation takes longer than usual, the system is equipped to respond quickly to the situation. JITbase collects real-time data and has shared its goals for solving an operator assignment model based on this data. It is worth noting that, however, this study is not directly focused on real-time simulations and related algorithm improvements.

This master's thesis focuses on studying an Operator Assignment Problem based on the requirements of our industrial partner. JITbase defines some non-identical parallel CNC machines, and some pre-sequenced tasks are already assigned to each CNC machine. Among the tasks assigned to each machine, some of them, named *not_lights_out,* need the supervision of an operator. JITbase provides optimal operator assignment solutions to allocate the operator to each CNC machine to supervise the *not_lights_out* tasks. In addition, JITbase has defined manual operations such as refilling the bar feeder, conducting quality control, and tool change, but JITbase software is not able to support the manual operations. Hence, this study provides a paradigm to assign one single operator to supervise the *not_lights_out* tasks and perform the manual operations. In other words, this study would solve the challenge and cover both the manual operations and regular duties together. Moreover, as mentioned, the tasks on each machine are pre-sequenced, but the manual operations are not pre-sequenced on each machine which makes the problem complex to solve. So, this study aims to tackle this challenge while achieving the shortest finishing time. Therefore, in this study, we identify the company's specific challenges and modify its model and assumptions accordingly.

The results of this study are not limited to our industrial partner but can be extended to several other companies in the manufacturing sector. The proposed model and results of this research can serve as a valuable reference for similar projects in other factories. By assigning a single operator to handle both tasks and replenishment operations, the manufacturing sector can reduce costs and improve efficiency. Given the ongoing worldwide labour shortage (Gružauskas, 2016), we are of the opinion that our model can help companies optimize the assignment of their limited human resources and boost the efficiency of their CNC machines and other automated machinery.

This study is performed in five chapters. First, the definition and a literature review are carried out to explain each concept in more detail and explore the existing methods and research.

In the second chapter, we introduce our industrial partner and provide a description of the problem, including an explanation of each task type and manual operation. We also delve into the CP model utilized by the JITbase, discussing its various parameters, constraints, variables, and objective functions. Furthermore, we highlight the shortcomings of the current model in terms of operator assignment to specific manual operations.

In the third chapter, we present the modifications made to the JITbase model to address the specific challenges and needs. We introduce new variables, parameters, and constraints to the original model explained in chapter two. The new optimization model is developed using CP.

In the fourth chapter, we implement this new model in an industrial setting using the Python language. The model can generate an optimal operator assignment plan for the upcoming production horizon in just a few minutes using real-time data. We develop the model to implement the operator assignment paradigm that addresses the challenges of both task supervision and replenishment in the company's production process.

In the final chapter, we summarize our work and provide recommendations for future research.

CHAPTER 1 LITERATURE REVIEW

This study focuses on developing a Constraint Programming model that generates the optimal operator allocation plan in response to JITbase requirements. The operator must supervise some regular tasks on CNC machines and additional manual operations, including for each CNC machine addition to regular tasks. Therefore, the challenge is to develop a model that efficiently assigns the operator to both the tasks and the manual operations. In the subsequent chapters, the model and implementations are explained in more detail.

In this chapter, we provide an overview of the main concepts used in this project, along with a review of the existing literature. First, we explain the concepts of scheduling and operator assignment problems, then, we delve into Constraint Programming and other concepts such as bar feeder and quality control.

1.1 Scheduling problem

Scheduling problems have attracted academic attention over the years. Santos et al. (2021) classified scheduling problems into single-machine, open shop, flow shop, and job shop categories. In addition, scheduling problems on parallel machines is another important category that is used not only in manufacturing but also in the applications of service sector or information systems (Yeh et al., 2014)

Our project involves a set of pre-determined tasks with a defined sequence. The goal is to assign an operator to machines to perform the tasks in their designated order while taking care of manual operations.

In the following, some studies related to scheduling problems are provided.

Graves (1981) defined production scheduling problems as the assignment of available resources in order to optimize some set of criteria. Rodammer et al. (1988) conducted a survey on production scheduling problems and proposed that the primary goal of a scheduling problem is to find a feasible way to assign resources to jobs while considering the defined constraints and minimizing production costs. They defined the resources as labour, material, and equipment. Turkcan et al. (2010) addressed a scheduling problem with non-identical parallel CNC machines. They developed a two-stage paradigm to minimize manufacturing costs and minimize total tardiness.

1.2 Resource assignment problem

The Resource Assignment Problem (RAP) is a specific optimization problem that involves assigning a set of resources to a corresponding set of tasks or activities in an optimal manner. These resources may include individuals, assets, products, materials, or capital. The RAP is commonly used in project management, scheduling, and logistics as the optimal allocation of resources is essential in achieving the problem objectives. The objective of RAP is to find an optimal assignment of limited resources to tasks to fulfill the problem objectives (Bouajaja and Dridi ,2017). Typically, the objective of the RAP is to maximize the profit or minimize the cost or time. The Human Resource Assignment Problem (HRAP) is also a specific type of RAP that focuses on assigning human resources, such as operators, workers, or employees, to various tasks or activities to minimize time or cost or maximize profit. HRAP is also referred to as manpower, worker, operator, and employee assignments. It should be noted that in this study HRAP is considered in a manner that the resource, which is the operator, assigns to some activities. These activities include some supervising tasks or some manual operations on machines.

Süer (1996) presented a two-phase approach to make an optimal decision on both cell loads and manpower assignment simultaneously. They considered a real manufacturing case and developed a Mixed Integer Programming (MIP) and an Integer Programming (IP) model. The goal was to find the optimal operator and product assignment. Bhaskar and Srinivasan (1997) considered an operator allocation problem for cell manufacturing in an assembly production environment. They developed a MIP formulation to minimize the $makespan$ and balance the workload among cells. They also defined static and dynamic versions of the problem and proposed mathematical formulations for them. Nembhard (2001) addressed a heuristic method for operator assignment. He considered learning rates for the operators. The method resulted in an improvement in overall productivity. Norman et al. (2002) generated a MIP for employee assignment in a cell manufacturing problem. The problem is aimed at maximizing the organization's effectiveness, which is defined as a function of output quality, productivity, and worker training cost. Lan and Lan (2005) addressed an HRAP model for the manufacturing sector and considered CNC machine productivity as the problem. The model aimed to optimize the production rate, holding cost, and production cost simultaneously. Pentico (2007) examined different types of assignment problems,

and he presented different assignment models, such as the classic assignment problem, the scheduling health care, and the transit problem. He specifically addressed personnel scheduling and assignment problem modeling. Süer and Tummaluri (2008) considered HRAP in a labour-intensive cellular environment. They accounted for the operator's skill level and implemented a three-phase method for the problem. The three phases include: the generation of the alternative configuration, computing cell sizes and cell loads, and assigning operators to the operations.

Yang and Chou (2011) presented a multi-objective model for manpower assignment. They considered the model for a consulting engineering firm to not only maximize the output but also create a balanced workload and reduce idle times. Also, to handle the complexity of the problem, they developed a particle swarm optimization algorithm.

Süer et al. (2013) investigated worker assignment and cell loading simultaneously and developed a two-step approach for that to minimize tardiness. In the first step, the manpower assignment is determined by using a mathematical formulation. Also, operator sharing between operations is allowed. In the second step, the problem focuses on cell loading. They concluded that sharing operators between operations yields a higher production rate.

Chen (2004) developed a model that considers resource allocation and job scheduling on parallel machines at the same time. They also considered costs for both the job schedule and resource allocation. They mentioned that the problem is NP-hard and provided a column generation-based branch and bound approach to cope with that. Kuo and Yang (2007) developed a MIP formulation for a multi-line operator assignment problem. They considered the problem from a case study. They also represented multi-skill operator allocation with the objective of minimization of the total skill sets assignment. Zaman et al. (2012) presented an operator allocation problem for an assembly line, considering the fact that the workstations are pre-determined. They also defined the objective as getting a sustainable balance line. Kuo and Liu (2017) proposed an operator allocation problem. In their case, the operator is allowed to have an inter-cell transfer. They developed two phases for the problem; in the first phase, the optimal assignment of tasks to workstations and assignment of the operator to the workstation is found by applying an IP approach. In the next phase, by using the results of the first phase, the problem minimizes the manpower required for the case. They

performed their methodology for a real case in Taiwan and showed that the methodology could improve manpower efficiency.

Bouajaja and Dridi (2017) considered HRAP as an optimal allocation of human resources to a certain number of tasks. They surveyed studies on HRAP from 1995 to 2017 and believed that the number of studies focused on HRAP after 2000 shows the fact that the personnel allocation concept has attracted attention in the academic community during these years. They classified the papers based on the case study in manufacturing, health care, project management, aircraft, maintenance management, and so on. They also examined different exact, heuristic, and metaheuristic solution methods for HRAP.

Santos et al. (2021) considered employee assignment and machine scheduling simultaneously in an industrial case study. They defined employee assignment as the optimal assignment of employees to supervise equipment operations. The authors presented an integrated approach, and they found an 11.6% improvement in the solution by considering the optimal personnel allocation.

Grillo et al. (2022) examined HRAP in Industry 4.0 and insisted on the importance of HRAP models in this context. They explained that the interconnection of personnel, object, and system is the basic factor of Industry 4.0, and the assignment of human resources as the personnel in such a system plays a crucial role. They addressed a new MILP mathematical model and implemented it in Industry 4.0.

1.3 Bar feeder, Quality Control, and Tool Change

As previously mentioned, there are some manual operations defined by JITbase. The manual operations include Bar feeders reloading, Quality Control (QC), and Tool Changes.

In manufacturing industries that use CNC machines, it is essential to provide the material for the machines. There are different ways of doing this, and one common approach includes a bar with a feeder named a bar feeder.

A bar feeder is thus a device used in manufacturing industries and machining operations to automate the process of feeding raw materials into a lathe or other machinery. With the presence of a bar feeder, the raw material is loaded into the feeder, which automatically feeds the machine as required. This removes the need for operator intervention to load raw material into the CNC

machine manually. Yet, some level of operator supervision is still needed to reload the bar feeder (Kasirolvalad et al., 2004). In Figure 1.1, an overall view of the bar feeder is depicted.

Figure 1.1 The Bar Feeder picture

Quality Control (QC) is a procedure that is used to control and guarantee that a product or service meets specific standards. It includes inspecting, testing, and evaluating products or services to ensure that they meet the required specifications. QC can be applied at different stages of the production process, from raw material to the final inspection of the finished product. The QC process for CNC machines mainly includes raw material inspection, calibration, and testing. QC in manufacturing systems can enhance the reliability of the product and reduce defects.

When machines and equipment are used for a certain period in a production line, they may stop working properly. To avoid this, a form of tool change has been implemented, which involves changing the tool on the machines before the occurrence of any potential breakdown.

As this study is mainly about providing a new paradigm of operator assignment to bring material for bar feeder, QC, and tool change, some studies related to this subject are also discussed in the following.

Benbouzid-Sitayeb (2013) addressed a production scheduling problem considering equipment maintenance. He proposed an integrated approach for maintenance tasks to develop practical schedules. He also considered manpower availability and human skills in his plan. Dang et al. (2014) presented a method for scheduling resources for feeders of the machine in a production line. They considered the mobile robot as a resource and developed MIP and a genetic algorithm for that. Their problem was solved for a real case in a production line.

Sheikhalishahi et al. (2016) considered multiple production lines and developed a multi-objective maintenance plan for that. In their study, human resources and budget are limited, and the production line must maintain a specific level of reliability. Bouzidi-Hassini and Benbouzid-Sitayeb (2013) considered a joint production and maintenance system and developed a multi-agent approach for scheduling the tasks. They also addressed the human availability and human skills in their study. Mathlouthi et al. (2018) addressed a multi-attribute operator scheduling and routing problem for maintenance equipment. In their problem, the technician must serve various maintenance tasks while satisfying some resource constraints. They developed a MIP formulation by considering some constraints such as technician skills, inventory, multiple time windows, and distance. Tortora et al. (2021) provided an extensive review of the maintenance issues caused by humans. The authors examined the studies in this area and concluded that there are still new opportunities and challenges in manufacturing and maintenance operations for human intervention. They mentioned that the operator plays a significant role in maintenance in the case of high-quality and safe operations.

1.4 Real-Time scheduling

As previously mentioned, our industrial partner uses real-time data for the model, which requires us to consider real-time systems planning. A real-time system can be challenging due to the dynamic nature of data. The system load can change over time, and new tasks can be removed or added at any time. Several studies have addressed this topic, which we outline below.

Mahdavi et al. (2010) addressed a dynamic cell manufacturing environment and developed an IP model for production planning and employee assignment. The authors considered some concepts simultaneously, multi-period production planning, workers' available times, machine capacities, and dynamic system reconfiguration. The goal is to minimize some costs related to handling costs, hiring and salary costs, and machine costs. Süer and Alhawari (2013) considered a dynamic multi-period cellular manufacturing environment and proposed two assignment strategies for that. They aimed to minimize the $makespan$ by considering the dynamic skill changes.

Reddy et al. (2018) studied a flexible $job\ shop$ problem with machine breakdowns. In their problem, the machine's breakdowns happen as real-time events. The authors developed a multi-

objective model to minimize the $makespan$ and minimize the machine load variation. Zhang and Wang (2018) considered a dynamic $job\ shop$ scheduling problem. They conducted the model on a case in an assembly manufacturing system and employed CP and a MIP formulation for that. Zhu et al. (2019) addressed a real-time $job\ shop$ scheduling problem in the assembly manufacturing industry with a multi-agent system and developed a novel adaptive method to solve that. Ghaleb et al. (2020) applied a real-time optimization model for maintenance and production. They considered three phases for the problem: first, they find the optimal preventive maintenance; then, they integrate it into a scheduling model; and finally, they extract the results from the scheduling model by giving the real-time information to it as input.

This study was conducted in collaboration with JITbase, a Montreal-based company that specializes in Industry 4.0 solutions for addressing operator shortage issues in production lines. JITbase considers some CNC machines with pre-defined tasks and a single human resource (operator). The operator is responsible for performing the tasks on all machines. The company develops solutions for optimizing the assignment of the operator to the tasks on each CNC machine.

The company tries to assign the least number of employees required for the CNC machines, reduce the machine downtimes, and increase the machine uptime simultaneously. They have three basic goals which are:

- Single operator, multiple machines: They plan to optimize the problem by enabling a single operator to manage several CNC machines simultaneously.
- Higher machine uptime: They aim to enhance the efficiency of CNC machines to achieve better overall uptime.
- More tasks with no need for the operator: They focus on utilizing analytics to improve the CNC programs, reducing the need for operator intervention in performing certain tasks.

The preliminary concepts and research for carrying out the project have been outlined.

JITbase had its own software solutions, but the solutions did not cover the manual operations such as QC, tool change, and bar feeder. So, JITbase asked an optimization model to support both the supervising tasks and manual operations. Thus, in order to better understand the problem definition, environment, and goals of JITbase, a series of meetings were conducted with the company's

engineers. During each meeting, various elements of the case study were discussed, allowing us to better understand the company's needs and the different processes. Subsequently, we refined the case study for further investigation.

CHAPTER 2 PROBLEM DESCRIPTION

The goal of this study is to assign a single operator to several CNC machines in an optimal manner to perform some defined tasks. In this chapter, we introduce the case study for the problem and describe the problem in detail. Then we explain the model with its parameters, variables, and constraints.

Before the arrival of automation in the production line, an operator was necessary to be always present on each machine. However, after automation, one operator became responsible for managing a set of machines. It means that the demand for the operator is now limited, but the optimal operation of the production line still requires human intervention to perform supporting tasks (Hamrol et al., 2018).

2.1 Problem overview

This study is defined based on the requirements of JITbase's clients to allocate their single operator to the specific tasks. Each client has information on machines, tasks, and other regulations. The goal of the problem is to identify the most efficient assignment of the operator to these tasks. The operator would see his plan on a monitor, which tells him the next machine and task to be performed. Figure 2.1 provides an overview of the CNC machine and data monitoring used in the company. Figure 2.2, which was published on the JITbase website, shows how the operator follows the optimal plan.

Figure 2.1 The company monitoring picture[1]

Figure 2.2 Operator plan[2]

[1] https://www.jitbase.com/

[2] https://www.jitbase.com/optimal-path-system

2.1.1 Tasks description

There are three types of supervising tasks defined by JITbase: $lights\_out$, $planned\ stop$, and $Stay\_on\_machine$:

- $lights\_out$: This task type does not require an operator to be performed as the machine is active for more than 5 minutes. This task type is indicated with a green light.
- $Planned\ stop$: This task type requires the operator's presence as the machine will be stopped for manual intervention, and the operator needs to be present. It is indicated with an orange light.
- $Stay\_on\_machine$: This task type requires the operator since the machine is active for less than 5 minutes, meaning that it needs human supervision. As a result, the operator stays on the machine and cannot perform other tasks during this time. It is indicated with a yellow light.

JITbase groups these types into two categories: $lights\_out$ and $is\_not\_lights\_out$. The $lights\_out$ tasks are those with the mentioned type $lights\_out$ that do not require operator intervention and can be conducted by the machine alone. On the other hand, $is\_not\_lights\_out$ tasks include $Planned\ stop$ or $Stay\_on\_machine$ tasks, which require the presence of the operator. Figure 2.3 provides an overview of the operator and tasks with their types on each machine in more detail. Based on this picture, the operator supervises the $Planned\ stop$ task with orange light, then he will supervise the $Stay\_on\_machine$ task with yellow light, but the $lights\_out$ tasks with green light works without needs of an operator.

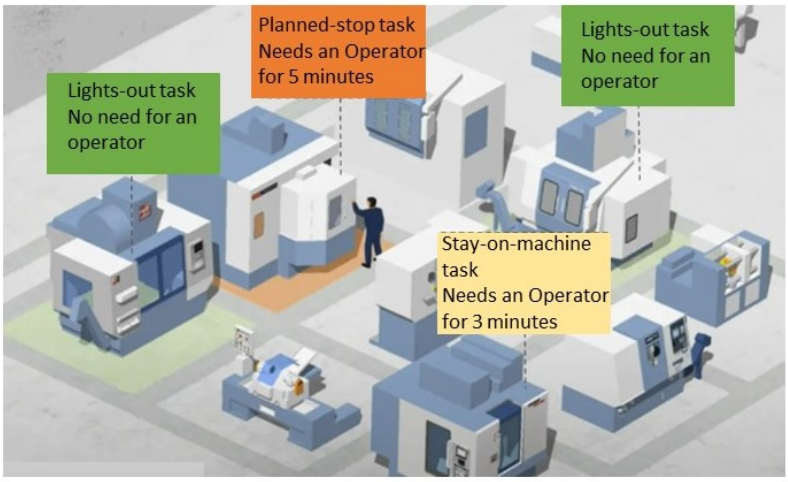

Figure 2.3 Task types in the problem

It should be noted that the sequence of tasks on each machine is fixed, and the problem's goal is to assign the operator to the pre-sequenced tasks on the CNC machines. In other words, the schedule of tasks on each machine is pre-defined by the company, but they need to determine how to allocate the operator to the machines to carry out the tasks.

In addition, the operator is responsible for conducting additional manual tasks defined by the company based on their specific requirements. In the next section, more detailed information is provided on this matter.

2.1.2 Manual operations

Robot arms or bar feeders are utilized to feed CNC machines, which reduces the need for human intervention. Despite this, an operator still must carry out parallel tasks to ensure that all machines remain active. The bar feeders need to be replenished after several tasks, meaning that the operator must supervise the bar feeders to ensure that they contain enough materials.

Moreover, there are other tasks that should be carried out by the operator. Once certain specific tasks are completed on each machine, the operator must conduct a tool change, as well as quality control. Thus, the operator must not only perform the pre-defined tasks on each machine but also these manual operations that are explained in the following paragraphs.

- Quality Control (QC):

Every characteristic of a CNC machine part must be measured by the quality control team. They typically inspect the machine parts and identify any components that require maintenance. The team should have comprehensive supervision of each machine to ensure that preventive maintenance is performed correctly. Certain characteristics of a machine may need to be checked every 15 parts, while others may only require inspection every 40 parts.

Therefore, the operator must conduct quality control for every x parts. It is considered a parallel task, and the machine should remain active.

- Reload Bar feeder or Robot:

  Every task performed on a machine requires a specific quantity of material. The material inside the machine is consumed after conducting certain tasks. Thus, the bar feeder must be refilled sometimes during the process by the operator.

  Additionally, each task on a machine consumes a particular amount of material. This information enables the company to calculate the part number at which a new material must be provided to a CNC machine.

  The operator must refill the bar feeder after every x parts to ensure that the machine continues to receive material during the process. It is also a parallel task, and the machine should remain active.

- Tool Change:

  Tool changes are considered as the replacement of a tool placed inside the CNC machine with a new one. Each task requires different tools to be performed; a specific tool may be changed after every 100 parts produced on the CNC machine, and another tool may be replaced after every 300 parts. Unlike other manual tasks, this is not a parallel task, as the machine needs to be stopped to complete this process.

The manual tasks are defined in our model as the reservoir actions. The operator must be present on each machine at the right time; otherwise, the machine will be turned off, and this must not happen while performing a task.

Therefore, the operator is responsible not only for carrying out the pre-defined tasks according to the scheduled "$lights\_out$" and "$is\_not\_lights\_out$" processes, but also for fulfilling the machine's requirements for material replenishment, tool changes, QC, and other manual tasks.

All these duties make the problem very complex to solve, but with the help of operation research, we can solve it efficiently. Table 2.1 summarizes the main types of tasks considered in the problem.

Table 2.1 Task types defined in JITbase.

	With pre-determined schedule	With frequency
CNC machine	$Lights\_out$ tasks	
Operator	$Is\_not\_lights\_out$ tasks	Tool change, Bar feeder, QC

2.2 Constraint programming

Rossi et al. (2006) defined Constraint programming (CP) as a powerful problem-solving approach that can model and solve combinatorial search problems. CP is a type of programming paradigm used in operations research, artificial intelligence, and computer science (Figure 2.4). A constraint in CP refers to a relationship between one or more variables, and the goal of CP is to identify a set of values to fulfill all the constraints.

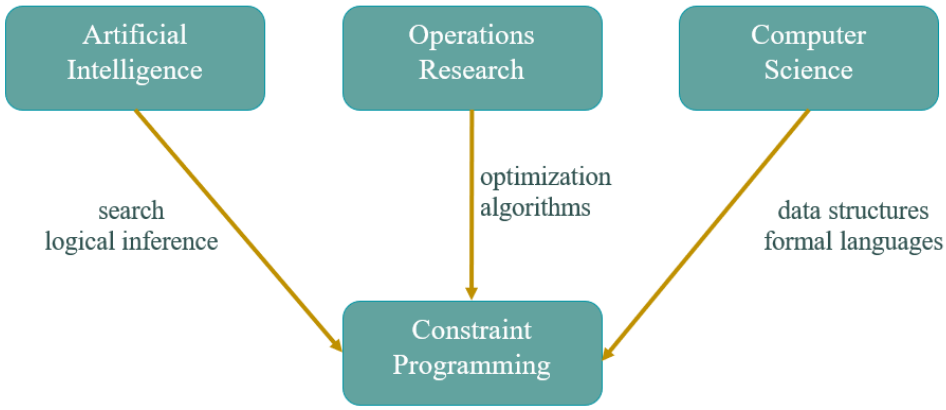

Figure 2.4 CP overview

The fundamental concept of CP includes dividing a problem into smaller subproblems, each of which can be modeled as a set of constraints. These subproblems can be solved either independently or in combination, using a diverse range of search and inference strategies to find a feasible solution. CP is highly advantageous for problems that contain complex, interrelated variables and constraints. It can be employed in a wide range of domains, such as scheduling, optimization, planning, and decision-making. CP models has advantages over MIP models:

- In CP, any expression over the variables is applicable.
    - e.g., $x3(y2 - e) \geq 12 + x2 \cdot min\,(x, y, z)$
- CP models can be more intuitive, close to natural language.
- CP applies a vide range of solving methods compared to MIP.

As mentioned, CP offers several advantages compared to integer programming, other notable of which are the use of global constraints, constraint propagation, and non-linear constraints (Walsh, 2002). Global constraints are advanced-level constraints that represent combinatorial structure and can be regarded as a combination of elementary constraints. Global constraints express constraints over multiple variables at once, unlike elementary constraints that deal with individual variables. This attribute makes them especially effective for modeling complex problems. For example, NoOverlap constraint is a global constraint that ensures that the variables do not overlap in time.

Overall, CP is a valuable tool in the field of operation research, particularly scheduling and assignment problems. The CP proposed in this study is based on Google OR-Tools[3], and it is solved by CP-SAT solver.

Chan and Hao (2002) demonstrated the effectiveness of the CP approach in production scheduling. Indeed, they developed a CP formulation to solve a scheduling model. Additionally, they outlined several methods for enhancing the computational performance of the CP method within their model. Finally, they compared the efficiency of CP with other heuristic methods in their problem.

[3] https://developers.google.com/optimization/reference/python/sat/python/cp_model

Yun and Gen (2002) used the CP technique for a pre-emptive and non-pre-emptive single machine *job shop* scheduling problem. They also proposed a novel genetic algorithm (GA) for the problem. Rodriguez (2007) developed a CP model for the scheduling and routing of trains. They applied the CP model to several real-case instances that were related to traffic in Paris. The model solution exhibited a substantial performance improvement. Moreover, the utilization of CP played a significant role in improving the flexibility of the model.

Zeballos (2010) addressed a flexible system in the manufacturing sector and developed a CP formulation for the scheduling of that. He also considered several characteristics, such as processing and set-up costs, the constraints on the number of tools in the system, the tools' lifetime, and the capacity of tool changer in machines. He achieved solutions with good quality by the CP approach. Berthold et al. (2010) presented a hybrid approach of integrating CP, IP, and satisfiability testing in an RCPSP problem. They considered some constraints such that the resource capacity must not exceed the makespan. They concluded that the hybrid between CP and satisfiability testing technique leads to the best approach.

Edis and Ozkarahan (2011) considered a problem with the parallel machine and developed a resource-constrained model with machine eligibility restrictions in an electrical appliance company. They also constructed IP and CP model and a combination of them for the considered problem. They took advantage of a problem-based search procedure for the combined IP/CP with efficient and quick results. Finally, they concluded that the combination framework of IP/CP performs perfectly for the investigated model. Novas and Henning (2014) considered resource-constrained flexible manufacturing systems (FMSs) and developed a novel approach in an integrated way. They developed a CP formulation in an integrated manner that considers all the sub-problems. The sub-problems include the scheduling, tool planning, allocation, and machine loading routing. They tested their CP formulation by comparing it to a problem with a different number of parts, tool copies, and various AGV speeds. Indeed, they emphasized the importance of machine allocation and task scheduling problems, tool management, and AGV issues.

Kreter et al. (2017) presented six various CP models for RCPSP, which aimed at minimizing the total duration. They considered calendar and temporal constraints for their model and concluded that CP solutions are extremely competitive with existing MILP model of the problem. Young et

al. (2017) developed CP to solve the multi-skill project scheduling problem. They implemented the model and tested it on 710 instances. Their CP model could solve 63% of instances within 10 minutes. Using the best of their configurations, they could close 87 open instances of the literature. Gökgür et al. (2018) considered a parallel machine environment and developed CP models for that. The models address the tool assignment and scheduling problems. The problem includes some parallel machines with several jobs that require a limited set of tools, and the purpose is to have the best assignment of the jobs and tools to machines while the $makespan$ is minimized. The authors considered three CP models and compared them with existing approaches. They used global constraints in some of the alternative models and stated that using the global constraint in CP makes the model easy to read and elegant. They compared the solution of CP with other mathematical programming methods, and they found that CP would result in more efficient solutions in a shorter time.

Fazel Zarandi et al. (2020) performed a review of novel scheduling problems and a comparison of CP with other approaches. Polo-Mejía et al. (2020) presented a MILP and a CP formulation for a multi-skill project scheduling problem in a real-case project. They considered a penalty for pre-emption, which means that each time an activity is interrupted, the penalty would be applied in the model. They obtained satisfactory results in a short time with CP.

2.3 Model description

The model is divided into two parts: the current model and the new model. JITbase developed a Python code related to the problem and sent it to us. Then, we extracted the base model from the code. Afterward, we added additional parts and modified the model. Finally, we coded the new proposed model and obtained results. This research aims to extend JITbase's model by adding new features to address these additional responsibilities.

2.3.1 Model assumption

Some assumptions must be highlighted in order to understand the model:

- There is only one operator available as the human resource.
- The tasks are non-preemptive, which implies that once a task begins, it must be completed.

- Each task must only be carried out on one machine. In other words, two machines cannot perform the same task at the same time.
- One task on a machine can only be performed when the previous task has been finished.
- The operator cannot be physically present on two machines at the same time.
- The sequence of tasks on each machine has already been defined by the company.

2.3.2 JITBase model

In the current model, the company only addressed the basic aspects of the problem and did not consider the manual tasks that are an integral part of the manufacturing process. We added those manual operations to the current model to make it more realistic and accurate.

CP, as explained before, is a powerful method for the industrial application of scheduling and production planning. In order to build the CP model, the following notations are used:

Index

m Machine

i, j Task

k Task type (is_light_out; k=1, $is\_not\_light\_out$; k=2)

Set and list

I Set of tasks

I_m Subset of tasks performed on machine m

M Set of machines

I^2 Subset of tasks with type "$is\_not\_light\_out$"

Parameters

H Planning horizon (in minutes)

OPR Number of operators

N Number of machines

β_j An immediate predecessor of task j on the same machine

r_i Number of operators needed for task i

d_i Nonnegative integer duration for task i

A CP model includes a set of variables, each with a defined domain, a set of constraints, and an objective function. This domain is a set of values, and in fact, a solution to the CP is an assignment of these values to variables in order to satisfy the constraints and optimize the objective function.

In CP, an interval variable is a decision variable whose values are represented as an interval $[s, e]$, where s is the start and e the end execution of task. In this problem, interval variables are used to represent each task. Therefore, defining an interval variable for each task would be the best way to assign the optimal start time. In this section, CP variables for the model are represented. Figure 2.5 provides a detailed illustration of interval variables.

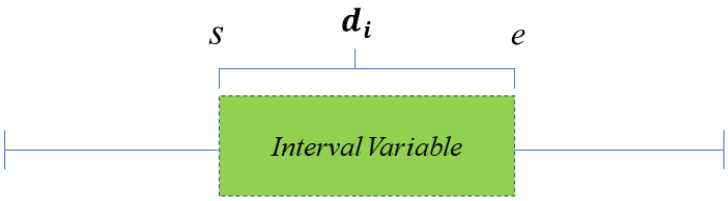

Figure 2.5 Interval variable definition

Also, as it is mentioned earlier, the sequence of tasks on each machine is fixed, so there is no need to define an interval sequence variable for this problem.

Variables

C_{max} Makespan

Interval variables

x_i An optional interval variable for task i, which includes:

- s_i: Nonnegative integer variable of the start time for task i
- e_i: Nonnegative integer variable of the end time for task i
- y_i: Boolean variable representing the presence of the interval variable.

A significant feature of an interval variable is that it can be optional, meaning that it may or may not be active in the solution. This concept is already captured in the definition of an interval variable, and no additional constraint is required. The only point is that we define a Boolean variable y_i to represent the presence or absence of the variable in the model.

Constraints

- Predecessor constraint:

$$s_{\beta_j} + d_{\beta_j} \leq s_j \qquad \forall\, m \in M\,,\, j \in I_m \qquad (1)$$

Constraints (1) indicates that each task is constrained to be completed before the start of the next task. This ensures that task j cannot start until its immediate predecessor has finished on each machine.

- No overlap constraint:

$$noOverlap\ (x_i, \mathrm{i} \in I_m) \quad \forall\, m \in M \qquad (2)$$

This constraint is a global constraint that ensures that each interval is forced to be finished before the start of the next interval, and all interval variables must have a non-overlapping sequence. In other words, this is a pre-defined constraint by which no two tasks on a machine can overlap. Figure 2.6 provides a clear illustration related to this constraint. It shows that two intervals x_3 and x_4 on machine M_1 overlaps together and violate constraint (2) and (1).

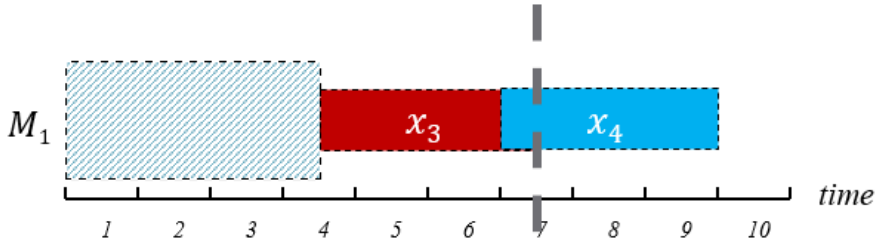

Figure 2.6 The NoOverlap constraint

Also, constraint (2) is not a necessary constraint because this constraint is already defined in constraint (1). The reason for the presence of this constraint is to improve the propagation. In other words, typically, in CP, standard propagation techniques are applied to each constraint separately, but global constraints consider those constraints as a group improving the resolution process of the model.

- The cumulative constraint for the operator:

$$Cumulative(\ (x_i)_{i\in I^2}\ , (r_i)_{i\in I^2}, OPR) \qquad (3)$$

The Cumulative constraint is also a global constraint in CP. It ensures that at any given time *t,* the total resource requirement (r_i) of the $is\_not\_light\_out$ tasks do not exceed the $capacity$ (OPR) of the resource. In this case, the resource $capacity$ is related to the number of operators, and the

resource requirement is related to the number of operators required for each task. It is worth noting that since we have one operator, this constraint can be defined as a $NoOverlap$ constraint.

Objective Function:

The objective function for this problem is to minimize the $makespan$, which is defined as the length of time for completing the last task. This objective function has been defined by JITbase, and we use it for the model.

Here, it is defined as the latest end time among all tasks on all machines. This is expressed as follows:

$$Min\ C_{max} \tag{4}$$

$$C_{max} = max(endOf(x_i)_{i \in I}) \tag{5}$$

Inequality (5) sets the variable C_{max} to the completion time of the last conducted task.

In this chapter, our industrial partner problem was introduced. Also, a detailed description of the company's operations and needs was provided. Then, the current model used by the company was presented, but it raises several questions that remain unanswered. For example, it does not address several issues, such as how machines are fed, the amount of material required for feeding the machines, how to assign the operator to both tasks and manual operations, number of times the operator should feed each machine. All the questions above are answered in the following section.

CHAPTER 3 METHODOLOGY

In this chapter, we introduce our proposed model and explain the new parameters, variables, and constraints. In this model, all the manual tasks are satisfied by one constraint named the Reservoir Constraint. We explain how the Reservoir Constraint works in the model with examples.

3.1 The model with parallel tasks

In this model, machines are fed by bar feeders that are refilled by an operator. As was mentioned before, this is important that the operator is aware of bringing the materials in a manner that ensures no bar feeder is left empty. Additionally, the operator is responsible for conducting other manual operations like tool change and QC.

Thus, it is necessary to define new tasks for refilling the bar feeders, changing tools, and controlling quality. In the proposed model, we define them as reservoir tasks.

In fact, we consider two task types in the new model:

1. Consumption tasks: The $is\_not\_lights\_out$ and $lights\_out$ tasks
2. Reservoir tasks: The bar feeder, QC, and tool change tasks

It should be noted that, unlike the consumption tasks that are pre-sequenced and pre-assigned to each machine, reservoir tasks are not fixed and sequenced on each machine. So, finding a solution to assign the operator to machines to take care of consumption tasks and reservoir tasks make the problem complex.

Moreover, we need to define new subsets and parameters for the reservoir part in the model:

Set and list

J Subset of tasks with type “reservoir"

J_m Subset of G_m optional tasks with type “reservoir” on machine m

$J_m{}^2$ Subset of reservoir tasks for tool change on machine m

Parameters

h_i Number of materials needed for consumption task i

$Cptm$ $Capacity$ of the reservoir of machine m

α A threshold defined for each reservoir

f_i Number of materials for reservoir task i

G_m Number of reservoir tasks available for machine m

g_m Upper bound on the number of reservoir tasks that could be performed on machine m

Also, it is necessary to modify constraint (3) since the operator should perform both the reservoir and $is\_not\_lights\_out$ tasks on each machine. Thus, the proposed model ensures that these tasks are carried out by the operator as below:

$$Cumulative\left((x_i)_{i\in I^2\cup J}, (r_i)_{i\in I^2\cup J}, OPR\right) \quad (6)$$

This constraint forces the model to ensure that at any point in time, the number of operators assigned for both reservoir and $is\_not\_lights\_out$ tasks must not exceed the number of available operators. Since we have only one operator in JITbase case, we should put $OPR = 1$.

In Figure 3.1, the cumulative constraint is depicted. As it is presented, one operator cannot perform two reservoir and $is\_not\_lights\_out$ tasks at the same time.

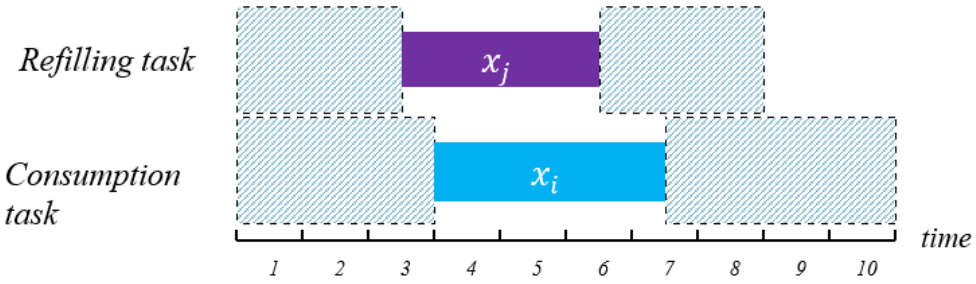

Figure 3.1 Example of an infeasible solution for the cumulative constraint on the operator

Furthermore, when a tool change is performed, the machine must be stopped. This means that no task must be performed during a tool change. Hence, the NoOverlap constraint must be changed as follows.

$$noOverlap\ (x_i, \mathrm{i} \in {J_m}^2 \cup I_m) \quad \forall\, m \in M \quad (7)$$

Besides, it is essential to ensure that machines never run out of materials and do not experience delays in QC. In other words, machines should never be turned off due to material shortage or late supervision. Therefore, it is critical to maintaining a minimum amount of material inside the bar

feeder and performing QC and tool changes on time. To satisfy these requirements, new constraints and equations are defined as reservoirs on machines:

$$AddReservoirConstraint\ (times, demands, min\_level, max\_level)$$

This constraint is a kind of cumulative constraint and based on the definition of Google OR-Tools, the reservoir constraint keeps the reservoir within a defined bound.

Additionally, according to the definition, this constraint ensures that at any $time \geq 0$, the reservoir level must be between min_level and max_level. Figure 3.2 illustrates a reservoir with the min_level and max_level, and based on this constraint, the material inside the reservoir must be within the min_level and max_level.

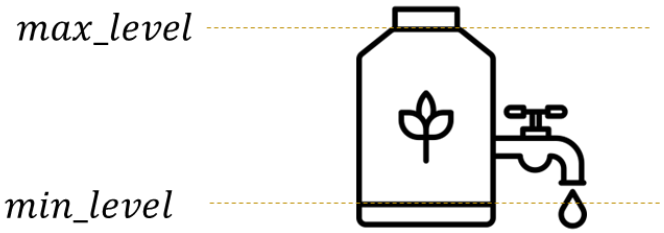

Figure 3.2 Reservoir constraint description

Each bar feeder can be fed as much as its $capacity$. In this constraint, the current model changes by "$demands$" if the variable "$times$" is assigned a value such as t. In this study, we use the reservoir constraint with active.

$$AddReservoirConstraintWithActive\ (times, demands, actives, min\_level, max\_level)$$

This constraint is like the one mentioned earlier, but the only difference is the "$actives$" part which are Boolean variables that represent which actions are actually conducted, meaning that the current model changes by "$demands$" if the variable "$times$" is assigned a value such as t and if "$actives$" is true.

Each element of "$time$", "$demands$", "$active$", "min_level", and "max_level" should be defined carefully in the model to be effective. We define each element based on our problem characteristics, and this constraint is written as below:

- The constraint for the reservoirs:

$$AddReservoirConstraintWithActive\big((s_i)_{i\in I_m} + (e_i)_{i\in J_m}, (-h_i)_{i\in I_m} + (f_i)_{i\in J_m}, (y_i)_{i\in I_m} + (y_i)_{i\in J_m}, 0, Cpt_m\big) \quad \forall\, m \in M \tag{8}$$

$$y_i = 1 \qquad \forall\, i \in I_m \tag{9}$$

As previously discussed, Constraint (8) maintains the inventory level of the reservoirs, ensuring that the material inside each of them remains within the minimum level and maximum level at any point in time. The variable "$times$" represents when the consumption happens. Also, "$demands$" is defined as the current inventory level of each reservoir and it represents the material inside each reservoir at any point in time. It changes when a task consumes materials ($(-h_i)_{i\in I_m}$) and the inventory level decreases by h_i, or when a refill action is completed ($(f_i)_{i\in J_m}$) and the inventory level increases by f_i. Therefore, the "$demands$" variable in this constraint is defined as $(-h_i)_{i\in I_m} + (f_i)_{i\in J_m}$

In addition, we need to define the "$active$" element; in this problem, the consumption tasks must always be performed, but the reservoir tasks must be performed when required. Therefore, a Boolean variable (y_i) should be defined to represent whether a reservoir task is performed by the operator or not. Hence, constraint (9) enforces the interval variable to be active for consumption tasks; however, for the reservoir tasks, the model needs to decide whether to perform them or not. In fact, this is equivalent to the following:

$$PresenceOf(y_i) = 1 \qquad \forall\, i \in I_m \tag{10}$$

Thus, the "$actives$" variable is defined as $(y_i)_{i\in I_m} + (y_i)_{i\in J_m}$, meaning whether a task should be performed or not; consumption task must always be conducted, and a reservoir task can be performed when required.

JITbase states that they consider min_level=0 for each reservoir, so we put this on the constraint. Finally, the last element is "max_level", which we put as the $capacity$ of each reservoir.

In addition, there are some critical parameters in the model which have a significant effect on the constraints and the solutions. They should be determined as accurately as possible to expect appropriate solutions for the problem. For example, we should recognize how many materials the operator brings. How many times should the operator go to bring the materials? How should we

define the subset for reservoir tasks? All the questions are answered by defining the equations below:

- The parameters for each reservoir:

$$g_m = \left\lceil \sum_{i \in I_m} \frac{h_i}{(1-\alpha)\,Cptm} \right\rceil \qquad \forall\, m \in M \qquad (11)$$

$$G_m = g_m(1+\alpha Cpt_m - min\_level)) \qquad \forall\, m \in M \qquad (12)$$

$$(f_i)_{i \in J_m} = [\,\lfloor (1-\alpha)Cpt_m \rfloor, \lfloor (1-\alpha)Cpt_m \rfloor + 1, \ldots, Cptm - (min\_level)]^{g_m} \qquad (13)$$

The three equations (11), (12), and (13) are related to the reservoir tasks and the number of materials needed for refilling. Based on what we defined in the "Parameters" section, α is the threshold of $capacity$, and each bar feeder cannot be replenished if the inventory level is higher than αCpt_m. Also, it is essential to know the number of reservoir tasks available for each machine. We define it as G_m and need it to define the set of reservoir tasks J_m. For better understanding, considering a pool of reservoir tasks of size G_m. To determine G_m, we need an upper bound g_m which is the number of copies of each reservoir task inside the pool. This number of copies is obtained by equation (11), which divides the material needs for all consumption tasks into the minimum threshold of the reservoir. Now, we can obtain G_m by the equation (12), which is equivalent to the number of g_m multiples of a certain number. This certain number is indeed the total number of items in a list that contains a series of numbers from small ($(1-\alpha)Cpt_m$) to large (Cpt_m - min_level) in order. Therefore, the number of items in this list comes from the difference between the lowest number of materials ($(1-\alpha)Cpt_m$) and the largest number (Cpt_m- min_level) for refilling, plus one for accounting. Also, for determining the number of f_i, we define equation (13). This is the list that we pointed out in equation (12) with the exponent of g_m. In fact, it is a list with the number of $\alpha Cptm$ with the exponent of g_m, $\lfloor \alpha Cpt_m \rfloor + 1$ with the exponent of g_m, etc. In other words, this is equivalent to g_m copies of αCpt_m, g_m copies of $\lfloor \alpha Cpt_m \rfloor + 1$, etc.

3.2 The reservoir constraints

In this part, some examples are provided to explain the equations in the model clearly. For a better understanding, Figure 3.3 and Figure 3.4 present these concepts in more detail.

Figure 3.3 displays the whole story. It shows the inventory level in the reservoir, refilling action, and consumption tasks. In this picture, the consumption tasks are depicted in blue color, and the reservoir tasks are shown in red color. We will delve more into this Figure after explaining Figure 3.4.

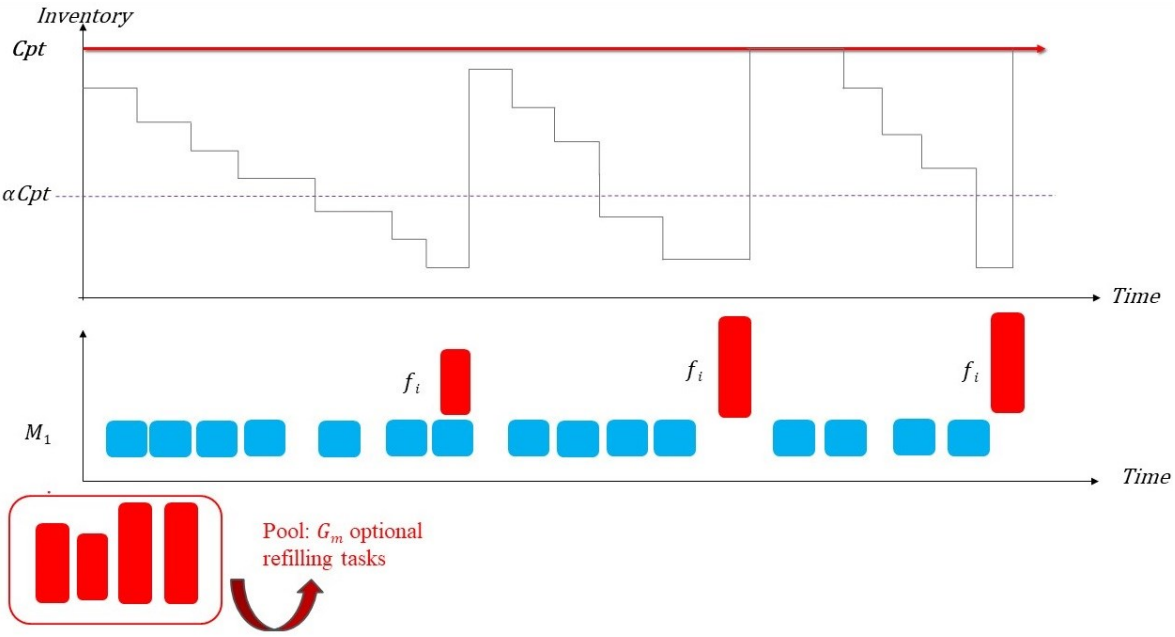

Figure 3.3 Refilling tasks description.

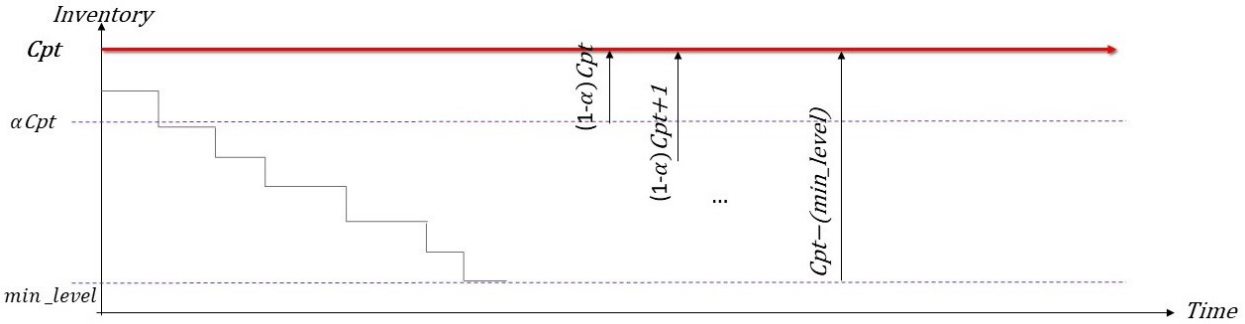

Figure 3.4 Inventory level with refill

As it is illustrated in Figure 3.4, the operator cannot perform refilling when the inventory level is between αCpt_m and Cpt_m. Also, the inventory level must not be under the min_level based on the constraint (8). Therefore, the operator must perform a refill when the inventory level is between

min_level and αCpt_m .In this manner, the material inside the reservoir does not exceed the Cpt_m, and the reservoir consumes without facing a shortage.

For example, we have 20 tasks for one machine, and a bar feeder related to this machine has $capacity = 10$, $\alpha Cpt_m = 8$, and $min\_level = 2$. Each of the tasks consumes one unit of the material, so we need 20 units of materials to perform all the tasks on the machine. Hence, g_m for this machine would be this material consumption over the minimum amount of refill as below:

$$g_m = \frac{20}{(10-8)} = 10$$

Additionally, we can calculate the number of materials that can be refilled:

$$(f_i)_{i \in J_m} = [\,(10-8), (10-8)+1, \dots (10-2)]^{10}$$

$$(f_i)_{i \in J_m} = [2,2,\dots,2,3,3,\dots,3,\dots,8,8,\dots,8]$$

This means that we have a list containing ten copies of two, ten copies of three…, and ten copies of eight. The operator has the choice to select each of them. We consider a task that needs material to be performed, but the reservoir does not contain enough materials! The operator must go and select each of the numbers mentioned according to the reservoir inventory level. For example, if the current inventory level is three, the operator can reload seven units or eight units.

Figure 3.3 clearly illustrates that when the inventory level reaches below the αCpt_m, the operator refills it by f_i. As mentioned, f_i is a list with the numbers eligible for refilling. Therefore, the operator refills the reservoir by a number in f_i list which is dedicated by the model. Additionally, a pool with red color is depicted in the picture, which shows all the available resource quantity for refilling. It should be noted that the pool size would be defined for the total horizon, and it will not change because it depends on the task's consumption and parameters of the reservoir, which are already given and will not vary in a horizon. For instance, when the inventory level goes below the αCpt_m for the first time, the operator goes to the pool, selects material, and refills the reservoir, then the tasks consume the material until the operator refills the reservoir for the second time. This time, he or she refills it at the largest number inside the pool, and the inventory level goes up to $capacity$.

Now that we have a clear understanding of these equations, we can explain how these parameters work on QC or tool change. For example, one CNC machine needs to be supervised for QC every 30-35 parts, meaning that the $capacity$ can be considered by 35 and the $\alpha Cpt_m = 5$ and $min\ \_level = 0$. Therefore, we have a list of numbers between 30-35 as below:

$$(f_i)_{i \in J_m} = [\,30, 31, 32, 33, 34, 35]^{g_m}$$

Accordingly, when the current inventory level becomes the $\alpha Cpt_m = 5$, or less, the operator would visit the machine for QC. For example, the operator would perform some jobs for QC when the inventory level is five, he or she conducts the job as he or she refills it for 30 units. It means that the machine does not need QC for the next 30 units. Figure 3.5 An example of QC shows an example of QC in this case to convey this concept clearly. This picture clearly illustrates that the operator performs QC task when the inventory level goes down to 5, and the machine will not need QC for the next 30 parts.

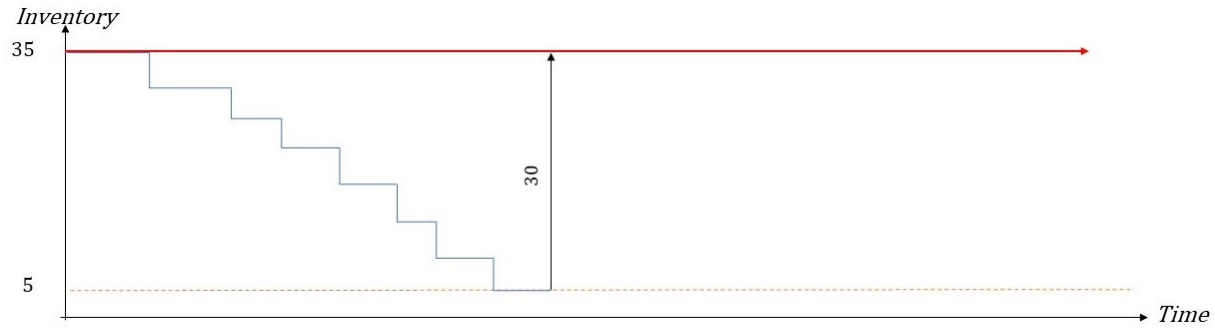

Figure 3.5 An example of QC

Another example could be a tool change that must be supervised every 300-305 parts. Thus, with the $capacity = 305$ and $\alpha Cpt_m = 5$, and $min\ \_level = 0$, the model can create a list between 300-305. The operator would visit the machine when the inventory level is equal to or less than $\alpha Cpt_m = 5$.

In this chapter, the reservoirs' model was presented in detail. The next step involves solving the model and analyzing the results. In the next chapter, we will present the results of the model with explanations.

As mentioned before, this study aims to implement the new features requested by JITbase. In Figure 3.6, the workflow diagram representing the planned implementation of the system is illustrated. We will delve into each step in this and the next chapter.

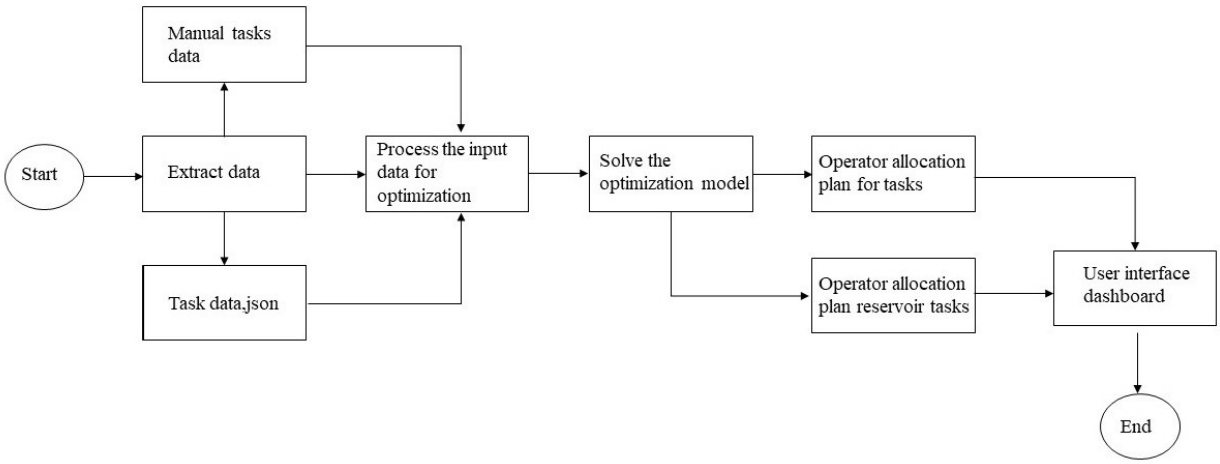

Figure 3.6 The project workflow diagram

3.3 Data extraction

The initial step is data extraction. Our industrial partner provided us with the data from their database. They sent two types of data: a JSON file containing information regarding the task schedules for each CNC machine and a file containing the data of the manual (reservoir) tasks, such as bar feeder reloading, tool changes, and QC. They also ordered the tasks on each CNC machine by their priority. The initial action for the project starts from this stage. We need to process the data using the two files and merge them into a single data.json file to serve as input for the problem. Table 3.1 addresses the prepared input data for the model. In this table, all the information that comes from the data is presented.

Table 3.1 Data description.

Tab	Description
Machine ID	List of all Machines IDs with their name
Task ID	Each CNC machine: list of all tasks IDs related to each machine with their name
Task Type	Each task: $lights\_out$ or $is\_not\_lights\_out$
Task duration	Each task: the duration of performing the task on the machine
Task consumption	Each task: the consumption of each task from the reservoirs (material, or the need for the quality control or tool change)
Reservoir tasks	Each CNC machine: a list of reservoir tasks such as tool change, QC.etc.
Reservoir tasks duration	Each CNC machine: processing time of the reservoir
Reservoir tasks capacity	Each CNC machine: capacity of each reservoir

Once we prepare the data file in the correct format explained before, we can solve the optimization model. We implement the model presented in Chapters 3 and 4 and then produce an assignment plan with two parts for tasks and reservoir tasks. The detail of the model is provided in Chapters 3 and 4.

3.4 User interface dashboard

Based on the solution of the optimization model, an assignment plan for the operator is ready to be used. Then, we create a user interface dashboard based on the assignment plan. This dashboard aims to present the upcoming operations to the operator on a screen in the company. The operator would be able to follow the dashboard to conduct the next action. In Figure 3.7 and Figure 3.8, we display screenshots of the dashboards mentioned above. The first dashboard displays the problem with machines and tasks without any reservoirs. The second dashboard presents both tasks and reservoir tasks, along with their planned release time in minutes. The two pictures are illustrated to represent the dashboards in detail and make a comparison between the scenarios with and without reservoir tasks.

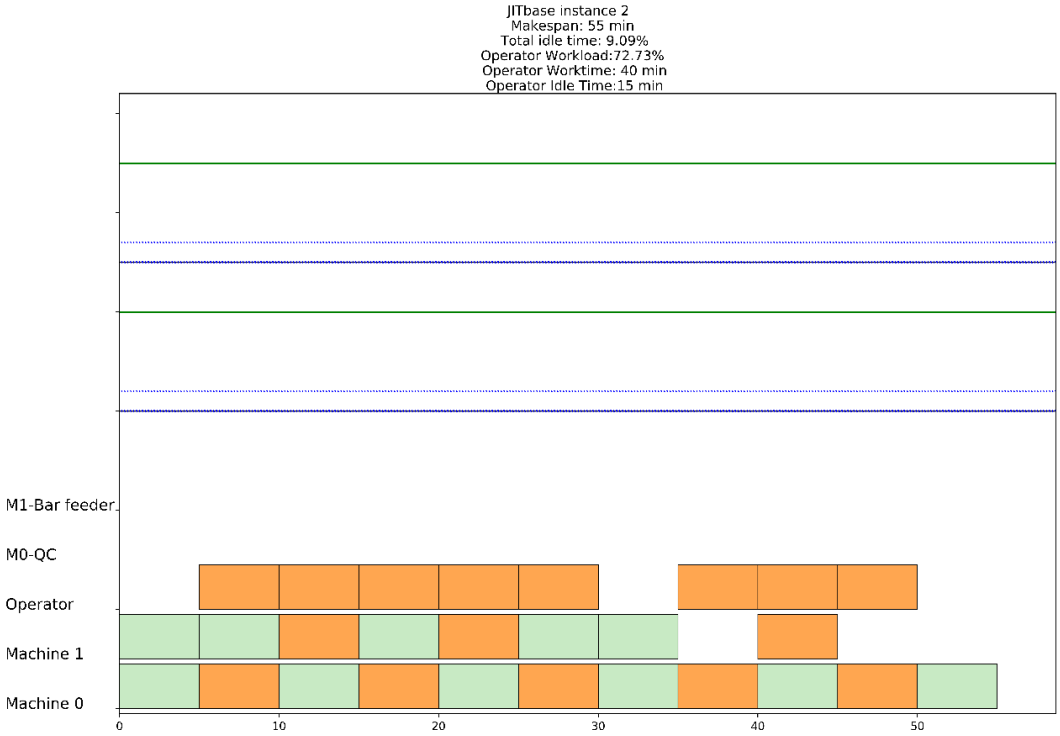

Figure 3.7 User Interface Dashboard without reservoir

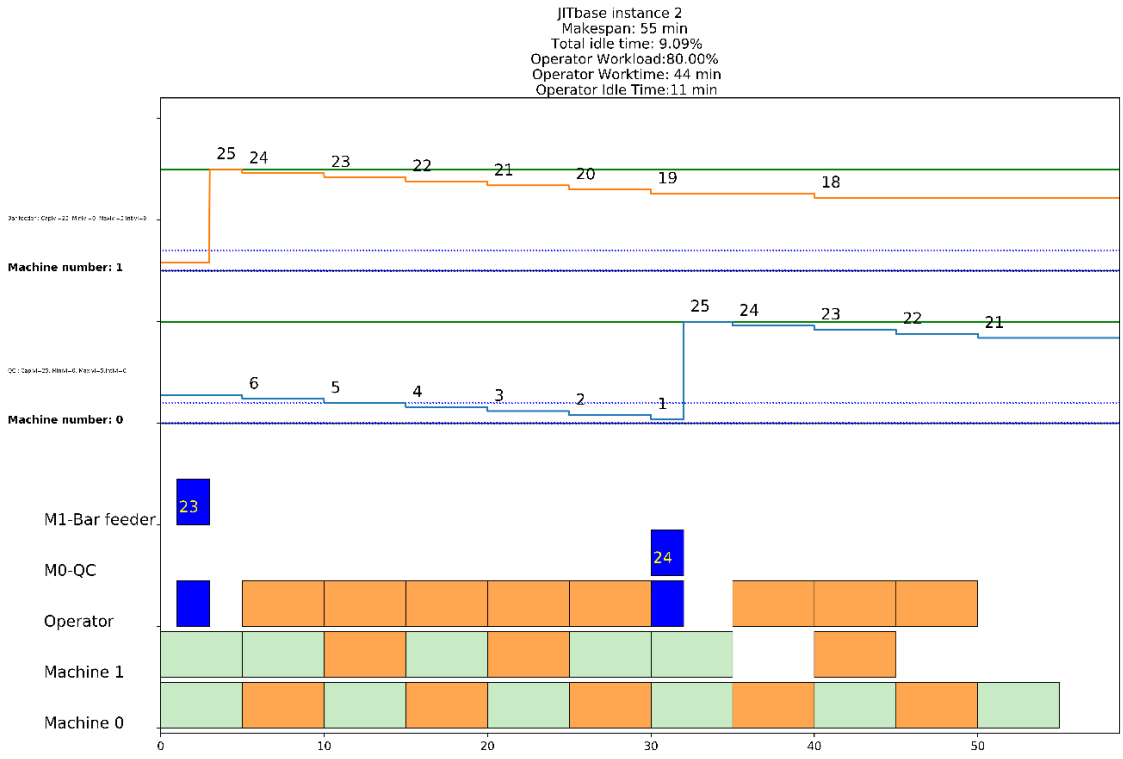

Figure 3.8 User Interface dashboard with reservoirs

In the dashboard, each CNC machine with its defined tasks is depicted. The optimal start time of the tasks on each machine is displayed as well. There is a specific line in the dashboard presented for the operator. The user (operator) can see which task or reservoir task needs to be performed at each instant. The blue tasks represent the reservoir tasks, the orange tasks represent the $planned\ stop$, and the green tasks illustrate the $lights\_out$ tasks.

Additionally, for a clear understanding of the inventory level in each reservoir, the Gantt chart displays the $capacity$, min_level, max_level, and the $initial\_level$, for each reservoir. It should be noted that the αCpt_m defined in this chapter is shown with max_level in the data set. The $initial\_level$ is the amount of material inside the reservoir at the beginning. The dashboard also displays the reservoir's inventory level as it is being refilled or consumed. The blue dot arrows indicate the max_level and min_level of each reservoir. Each number on the inventory line represents the current amount of material inside the reservoir. For example, as Figure 3.8 illustrates, the operator must first refill the reservoir “bar feeder” on the second CNC machine, and then he must go to the first CNC machine to conduct the orange task. Afterward, he must carry out the

orange task on the second machine. For the second machine, the inventory starts from eight units ($initial\_level = 8$), then the task consumes it until the inventory level goes below the ($max\_level = 5$), so the operator goes to the bar feeder and refills it by 23 units.

In addition, the company can find the $makespan$, operator idle time, and machines idle time in an optimal manner just by looking at the dashboard. On the top of the picture, this information is presented. Total idle time represents the idle time of all machines in the instance. The reason for the name is that the company named them like this, and we did not change it. Additionally, the operator's idle time shows the time that the operator is free.

We analyze some factors in the results of the problem. Two factors named “Operator Idle Time” and “Machine Idle Time" are essential factors for JITbase.

The price of each CNC machine is so high that the company desires the lowest machine idle time. On the other hand, they intend to use the operator as optimally as possible. Hence, when we analyze the results, it is better to consider these two factors as well.

Moreover, when we compare Figure 3.7 and Figure 3.8, we see that the reservoir tasks are conducted in parallel with other green tasks. It means that without the need for a second operator to perform the reservoir tasks, one operator can carry out both the tasks and reservoir tasks without an increase in $makespan$ and total idle time. Furthermore, when the operator is performing the blue tasks, he or she cannot conduct any orange tasks, which we have already defined before in constraint (7).

All in all, the dashboard is very useful for both the operator and the company managers. The operator would understand his or her task allocation during a $horizon$, and he or she knows how much material is required for the refilling of each machine. Moreover, the company would be able to find important information just by looking at the dashboard. The managers can also find the answer to some questions below:

- When are all the tasks finished ($makespan$)?
- What is the operator's workload? Is the idle time for the operator more than usual? Or is the operator overwhelmed by tasks?
- What is the total machine idle time?

- What is the current inventory level of materials at each moment?
- How much material should the operator bring each time?

CHAPTER 4 COMPUTATIONAL RESULTS

4.1 Sensitivity analysis

In this section, the results of instances from JITbase are provided to see how the model will optimize the problem. Also, we will have several analyses of some instances in the following.

Before analyzing the results, there are some indicators that need to be defined. The relevant notations for definition are addressed below:

Notations:

TIT	Total idle time
IT_m	Idle time of machine m
TIT_C	Total idle time as a ratio of makespan
OIT	Operator idle time
OW	Operator workload
OWT	Operator working time
OE	Minimum operator work time
D_m	Machine's processing time
$\overline{D}$	Average machine's processing times
$\overline{IT}$	Average machine idle time
R	Range of machines' processing times as a ratio of $\overline{D}$

Also, we need to clarify what is the "operator workload," which means the workload of the operator after solving the model which comes from this formulation:

$$OW = \left(\frac{OWT}{C_{max}}\right) \tag{14}$$

For machine idle time, it would be:

$$IT_m = C_{max} - (\sum_i d_{i,m}) - (s_1) \qquad \forall\, m \in M \tag{15}$$

Also, we have:

$$TIT = \sum_{m} IT_m \tag{16}$$

$$TIT_C = \frac{TIT}{C_{max}} \tag{17}$$

$$\overline{IT} = \frac{TIT_C}{M} \tag{18}$$

Since the indicators required for analysis are defined, we will delve into the JITbase data, the model results, and the analysis. Although the “operator workload” is obtained from solving the model, the "min operator engagement" depicted by OE is obtained before solving the model. It is defined as the minimum engagement time of the operator; for example, we know that the orange tasks must be performed by the operator and we are also aware of the processing time of each orange task, so we can compute this indicator to have an estimation of operator minimum engagement. It will be calculated as the processing time of all orange tasks over the average processing time of all tasks on all machines. The purpose of this indicator is that the minimum operator workload should not be more than 80-85% for each instance. Because the operator must perform other reservoir tasks, and if he or she is very busy with the tasks, we will see a high rate of machine idle time! And as we mentioned earlier, the machines' idle time would cost a lot for the company! Thus, this indicator will show if the operator has enough time to perform reservoir tasks without increasing the machines' idle time more than normal. Hence, this indicator is utilized to estimate how much the data is balanced in case of operator idle time and the machine idle time.

This indicator is defined as below:

$$D_m = \sum_I d_{i,m} \qquad \forall\, m \in M \tag{19}$$

$$\overline{D} = \left(\frac{\sum_m \sum_I d_{i,m}}{M}\right) \tag{20}$$

$$OE = \frac{\sum_m \sum_{I^2} d_{i,m}}{\overline{D}} \tag{21}$$

We also bring the "range" indicator, which means the difference between the maximum and minimum processing times of all machines. This indicator reveals the balance of machines; for instance, if we have one machine with a total processing time of 20 and another machine with a total processing time of 200, this data is unbalanced and makes the related instance easier and not realistic.

$$R = \frac{\left[\max_m D_m - \min_m D_m\right]}{\bar{D}} \tag{22}$$

In the following, we are planning to compare the computational time, $makespan$, and operator workload in case of increasing the number of machines, number of reservoirs, and number of tasks. The instances related to them come from the data sent by JITbase with the difference that we manipulate some features in the data to analyze various situations.

It should be noted that in each table, there are nine columns: the example number, the number of machines, the number of tasks per machine, the number of reservoirs per machine, the $makespan$, computational time, operator workload, range, and minimum operator engagement.

In addition, each example of information is the average of five instances, as mentioned earlier.

The first table for test analysis is Table 4.1, which represents the change of the parameters in a situation where the number of tasks and reservoirs are fixed, and we increase the number of machines. It contains two examples with 14 tasks and three reservoirs per machine. The difference between the two examples is the number of machines, as mentioned. As noted before, each example is the average of five instances. For example, information of example number one in Table 4.1, is the average of five instances with two machines, 14 tasks, and three reservoirs.

As Table 4.1 displays, with the increase in the number of machines, the computational time, the $makespan$, and the operator workload increase, this is because the operator must perform more tasks and attend more reservoirs for the three machines compared to two machines.

Table 4.1 Impact of a change in the number of machines.

example	machines	tasks per machine	reservoirs per machine	C_{max} (Min)	CPU (Sec)	OW	R	OE
1	2	14	3	114.8	0.79	72.42%	4.80%	62.58%
2	3	14	3	115.6	1.13	81.11%	7.28%	72.88%

Table 4.2 represents the impact of number of tasks on computational time, $makespan$, and operator workload. All the examples in this table have two machines and four reservoirs per machine. The first vivid point that comes from this table is that with increasing the number of tasks per machine, the $makespan$, computational time, and operator workload will increase. The second

point is that the changes in computational time are more evident than the changes in *makespan* and operator workload. It is better to display the computational time changes in a chart.

Table 4.2 Impact of a change in the number of tasks for instances with two machines.

example	machines	tasks per machine	reservoirs per machine	C_{max} (Min)	CPU (Sec)	OW	R	OE
1	2	14	4	107.8	0.85	72.26%	5.33%	60.77%
2	2	30	4	240.2	4.44	74.29%	10.58%	73.87%
3	2	50	4	430.2	36.42	74.56%	11.90%	64.60%
4	2	60	4	465.6	56.18	74.64%	7.88%	65.73%
5	2	80	4	639.4	172.48	80.74%	4.96%	70.25%

For a better understanding of the situation, Figure 4.1 is provided to show the changes in the computational time in more detail. As it is displayed, the slope of the graph between 60 and 80 is greater than other areas, which means that with 60-80 tasks, the computational time would increase more than for values smaller than 60.

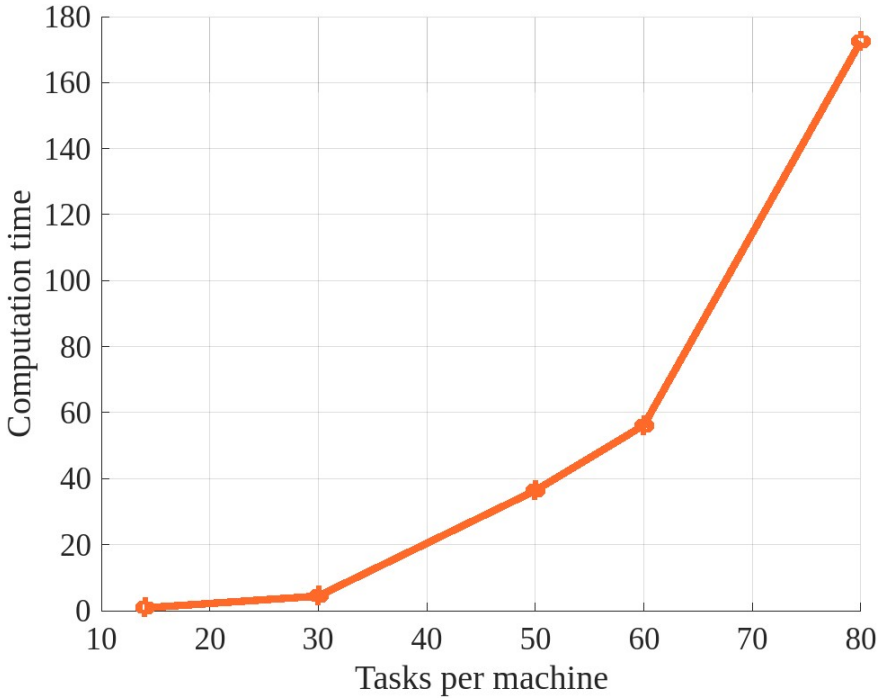

Figure 4.1 Impact of the number of tasks on the computational time with two machines

Table 4.3 is like Table 4.2 with a difference in the number of machines. In this table, we have three machines, and we are interested in investigating the changes on three machines. Like Table 4.2, when we increase the number of tasks per machine, the $makespan$, computational time, and operator workload will increase. Also, Figure 4.2 is provided to show the changes in computational time in more detail.

When we compare Figure 4.2 to Figure 4.1 we see that with the increase of machines, the model is not able to solve instances with 80 tasks. Also, for the same number of tasks, the $makespan$, computational time, and operator workload increase. For example, considering the example with 50 tasks and four reservoirs, the first example with two machines has the lower figures for $makespan$, computational time, and operator workload compared to the same example with three machines. In Figure 4.2**,** the effect of increasing the number of tasks on computational time is depicted. Based on Table 4.3, with the increase of tasks per machine to 80 tasks, the computational time increases in such a manner that the model is able to find a solution in five minutes (300 seconds). It should be pointed out that five minutes is the maximum amount the company considers for the instances.

Also, in Figure 4.2, a comparison between the computational time of the two tables is presented. As it is illustrated, with the increase in the number of the machines, the computational time increases. Figure 4.3 displays same style for both the blue figure with three machines and orange with two machines; both of them have considerable increases in 50 and 80 tasks.

Table 4.3 Impact of a change in number of tasks for instances with three machines.

example	machines	tasks per machine	reservoirs per machine	C_{max} (Min)	CPU (Sec)	OW	R	OE
1	3	14	4	112.4	1.13	82.59%	6.22%	77.42%
2	3	30	4	250.6	6.91	84.67	5.25	78.96
3	3	50	4	432.6	52.74	84.93	6.05	75.98
4	3	60	4	472.2	85.75	86.25	5.93	80.28
5	3	80	4	No solution in 300 Sec	No solution in 300 Sec	No solution in 300 Sec	No solution in 300 Sec	No solution in 300 Sec

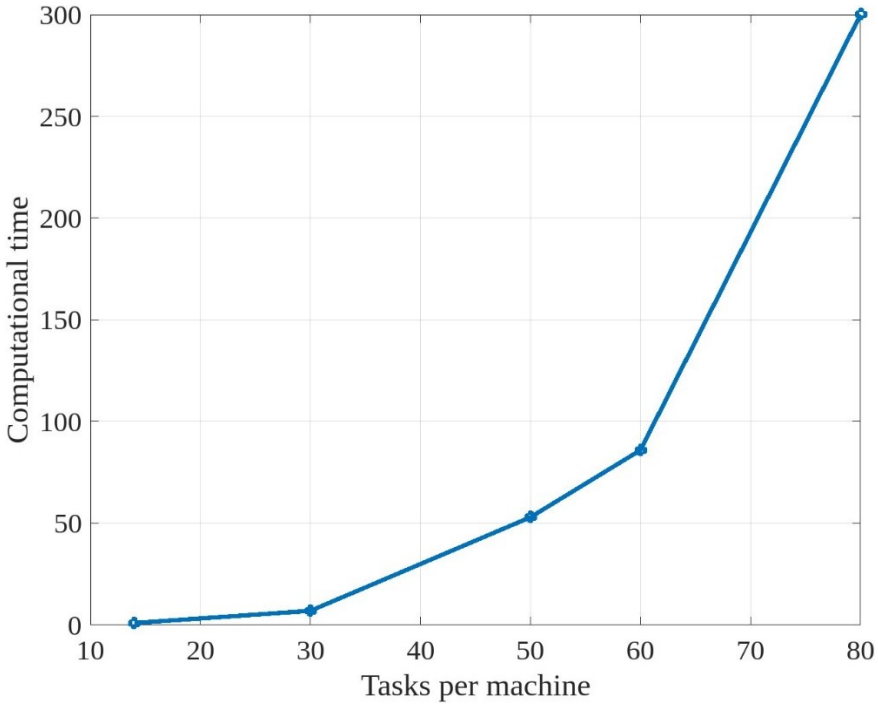

Figure 4.2 impact of the number of tasks on the computational time with three machines

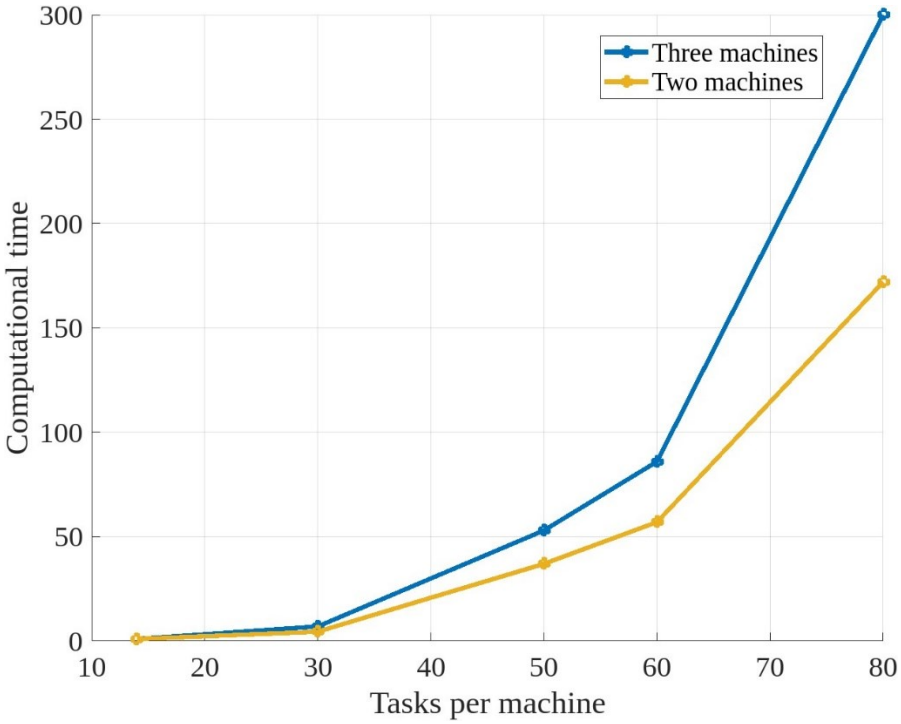

Figure 4.3Comparison of the computational times with two and three machines

In Table 4.4, we see the impact of the change in the number of reservoirs in examples with two machines and 20 tasks per machine. We are interested in investigating the impact of changing the number of reservoirs on the indicators. With an increase in the number of reservoirs per machine, the computational time, $makespan$, and operator workload increase. Among these indicators, we see more changes in computational time as well.

Table 4.4 Number of reservoirs changes with two machines.

example	machines	tasks per machine	reservoirs per machine	C_{max} (Min)	CPU (Sec)	OW	R	OE
1	2	20	3	154.6	1.41	63.96%	8.41%	54.48%
2	2	20	4	155.4	2.56	65.97	6.88	53.53
3	2	20	6	159.4	2.69	69.58	5.53	57.75
4	2	20	8	159.8	3.18	74.02	6.60	57.15
5	2	20	10	163.8	4.02	76.69	6.78	68.10

Table 4.5 is like Table 4.4, with a difference in the number of machines. In this table, we are interested in exploring the changes observed when there are three machines. Like in Table 4.4, with an increase in the number of tasks per machine, the $makespan$, computational time, and operator workload increase. Also, Figure 4.4 is provided to exhibit the changes observed in more detail.

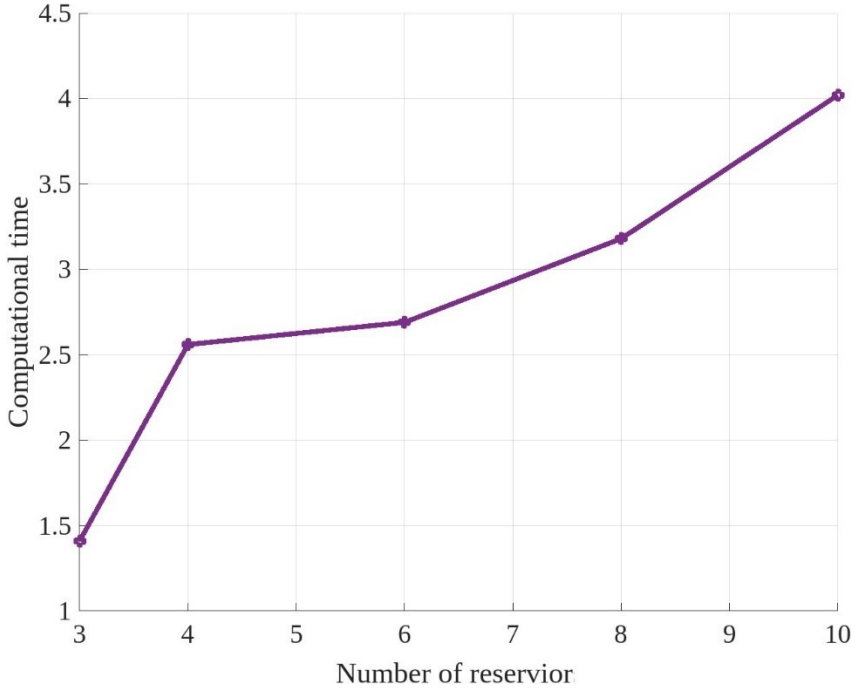

Figure 4.4 Impact of the number of reservoirs on the computational time with two machines

Again, Figure 4.5 is provided below to present the detailed changes in computational time. As it is displayed, when the number of reservoirs increases to ten, the computational time will explode! But still, the time is less than 300 seconds, and the operator workload is reasonable. Hence, we can conclude that, in this case, the company will be able to assign more reservoirs, even ten, to the operator without disturbing the operator workload and the computational time.

Table 4.5 Impact of the number of reservoirs on the computational time with three machines

example	machines	tasks per machine	reservoirs per machine	C_{max} (Min)	CPU (Sec)	OW	R	OE
1	3	20	3	155	1.45	76.50%	9.01%	72.06%
2	3	20	4	155.6	2.56	79.18	6.22	72.23
3	3	20	6	160.2	4.60	83.57	5.38	72.75
4	3	20	8	160.6	7.54	87.95	6.18	71.04
5	3	20	10	164.2	125.40	88.60	5.23	67.24

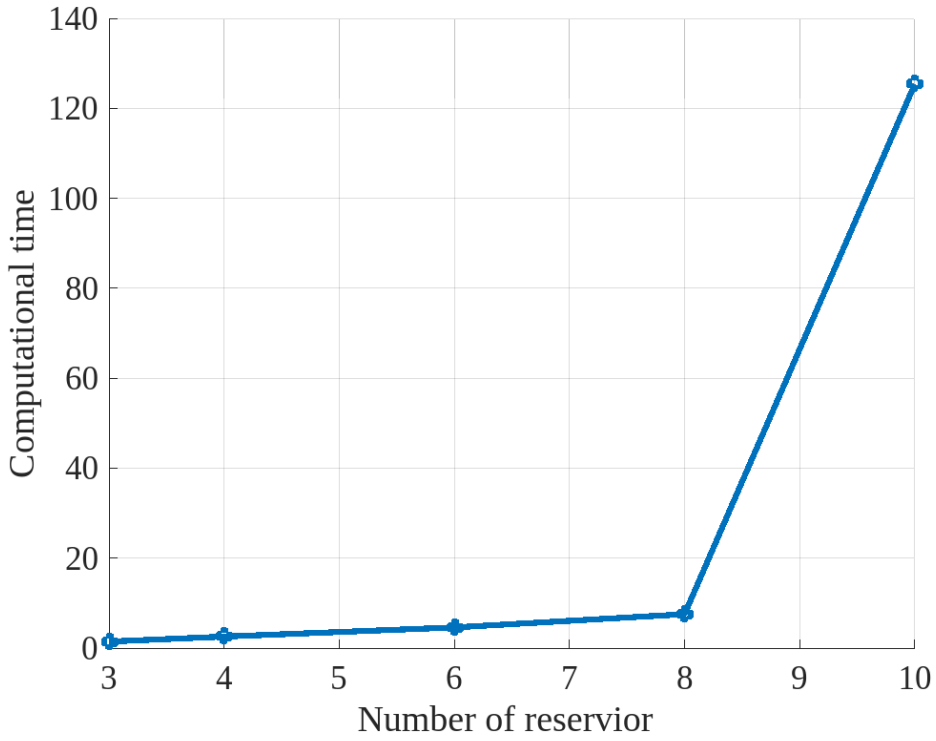

Figure 4.5 Change the number of reservoirs on computational time with three machines.

The interesting point from our analyses is that the number of tasks per machine has the most impact on computational time. It means that with an increase in the number of tasks per machine, the computational time will change more than with a change in the number of reservoirs or the number of machines.

4.2 Results on JITbase data

The company sent us two files, one for the data for tasks and machines, and one for the reservoirs. We processed them into appropriate data for the model; then, we solved the model for these instances. All the instances in this study are solved with Google OR-Tools CP-SAT solver and CP-SAT model on a computer with 11th Gen Intel® Core (TM) i5-1135G7 @ 2.40 GHz. JITbase sent us information on 17 reservoirs for each machine with one "QC", one "bar feeder", and 15 "tool changes" for each CNC machine. Each reservoir is specified by its $capacity$, max_level, min_level , and $initial\_level$. Table 4.6 represents the results of four instances with JITbase data. The first example is related to three machines with 50, 48, and 48 tasks respectively. The results collected after solving the instances are reported in Table 4.6. The computational time for solving

this instance is 300 seconds. Accordingly, the operator workload is 81.7 %, and the average idle time per machine is 19.4 %. This means that the operator is busy 81.7 percent of his time, and each machine is free for 19.4% during the horizon. Additionally, example number two is related to two machines, each of which has 17 reservoirs and 50 and 48 tasks respectively. This instance was solved faster with less operator workload and less machine idle time. The reason behind these results is that we have one machine less compared to the previous example with three machines, which means fewer tasks and reservoir tasks to be supervised. Additionally, since the operator has more free time to supervise the machines, we expect less machine idle time for the instances with two machines compared to three machines.

Table 4.6 Results of JITbase instances.

example	machines	tasks per machine	reservoirs per machine	C_{max} (Min)	CPU (Sec)	OW	R	OE	$\overline{IT}$
1	3	50-48-48	17-17-17	459	300.2	81.7%	3.4%	70.8%	19.4%
2	2	50-48	17-17	448	104.5	53.18	6	52.3	3.5
3	2	50-48	17-17	447	145.4	57.05	2.6	55.4	6.7
4	2	48-48	17-17	439	49.6	57.4	5.5	53.1	6.8

In this chapter, the results of the instances solved by the model were presented. These results are very vital for the company to recognize how they can effectively assign their operator to have the minimum $makespan$, idle time, and the model computational time. Moreover, we compared some examples with different task numbers, reservoirs, and machine numbers. We also explained that the number of tasks is one of the most important parameters with respect to computational time.

CHAPTER 5 CONCLUSION AND RECOMMENDATIONS

5.1 Summary

We improved and solved the model of human resources allocation in the manufacturing sector for JITbase software. The model is an optimization model based on constraint programming. The company is using an optimization model for assigning a single operator to machines, but their existing model was not able to satisfy all their needs. Based on the company's desires, some features were added to the base model. One issue that has been an important challenge for JITbase is refilling the machines. They defined their problems with one operator, and they required a model to allocate this limited human resource to machines to perform the tasks as well as refill the materials inside each machine.

The proposed optimization model succeeds in solving instances within a reasonable computing time. Also, some analyses were provided to examine the behaviour of the model in situations where the number of tasks, machines, and reservoirs increases. The results of these analyses would help JITbase's clients to decide the best solution for them. Moreover, a comprehensive dashboard was created that shows all the information in one picture. With this dashboard, the operator can understand his next task to perform, and the company can handle the situation by knowing the current inventory level of material inside each machine, the operator assignments, the operator workload over the horizon, the $makespan$, and all the machines' idle time.

We believe our model would help the JITbase in optimizing the use of their available machines and human resource and boost the productivity of their automated equipment. This model could also be helpful for systems that face a shortage of human resources.

5.2 Future research

For future research, it would be interesting to see how the model would behave if we have more than one operator. In addition, it would be nice to consider operator skills in the problem. Under this assumption, the operators could be allocated to tasks according to their various sets of skills. One also develops an algorithm to decide which operator assign to each machine.

REFERENCES

Berthold, T., Heinz, S., Lübbecke, M.E., Möhring, R.H., Schulz, J. (2010). A Constraint Integer Programming Approach for Resource-Constrained Project Scheduling. In: Lodi, A., Milano, M., Toth, P. (eds) Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. CPAIOR 2010. Lecture Notes in Computer Science, vol 6140. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-13520-0_34.

Bhaskar, K., & Srinivasan, G. (1997). Static and dynamic operator allocation problems in cellular manufacturing systems. *International Journal of Production Research*, *35*(12), 3467-3482.

Bouajaja, S., & Dridi, N. (2017). A survey on human resource allocation problem and its applications. *Operational Research*, *17*(2), 339-369.

Bouzidi-Hassini, S., & Benbouzid-Sitayeb, F. (2013, April). Multi-agent based joint production and maintenance scheduling considering human resources. In *2013 5th international conference on modeling, simulation and applied optimization (ICMSAO)* (pp. 1-5). IEEE.

Chan, W. T., & Hu, H. (2002). Constraint programming approach to precast production scheduling. *Journal of Construction Engineering and Management*, *128*(6), 513-521.

Chen, Z. L. (2004). Simultaneous job scheduling and resource allocation on parallel machines. *Annals of Operations Research*, *129*(1), 135-153.

Dang, Q. V., Nielsen, I., Steger-Jensen, K., & Madsen, O. (2014). Scheduling a single mobile robot for part-feeding tasks of production lines. *Journal of Intelligent Manufacturing*, *25*(6), 1271-1287.

Edis, E. B., & Ozkarahan, I. (2011). A combined integer/constraint programming approach to a resource-constrained parallel machine scheduling problem with machine eligibility restrictions. *Engineering Optimization*, *43*(2), 135-157.

Fazel Zarandi, M. H., Sadat Asl, A. A., Sotudian, S., & Castillo, O. (2020). A state of the art review of intelligent scheduling. *Artificial Intelligence Review*, *53*(1), 501-593.

Ghaleb, M., Taghipour, S., & Zolfagharinia, H. (2020, January). Real-time optimization of maintenance and production scheduling for an industry 4.0-based manufacturing system. In *2020 Annual Reliability and Maintainability Symposium (RAMS)* (pp. 1-8). IEEE.

Gökgür, B., Hnich, B., & Özpeynirci, S. (2018). Parallel machine scheduling with tool loading: a constraint programming approach. *International Journal of Production Research*, *56*(16), 5541-5557.

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

Grillo, H., Alemany, M. M. E., & Caldwell, E. (2022). Human resource allocation problem in the Industry 4.0: A reference framework. *Computers & Industrial Engineering*, *169*, 108110.

Gružauskas, V., Karosevičiūtė, D., & Srovnalíková, P. (2016). Labour and machine efficient utilization importance to the enterprise profit. *Journal of Management*, (1), 28.

Hamrol, A., Zerbst, S., Bozek, M., Grabowska, M., & Weber, M. (2018). Analysis of the conditions for effective use of numerically controlled machine tools. In *Advances in Manufacturing* (pp. 3-12). Springer International Publishing.

Kasirolvalad, Z., Motlagh, M. J., & Shadmani, M. A. (2004). An intelligent modular modelling approach for quality control of CNC machines product using adaptive fuzzy Petri nets. In *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference, 2004.* (Vol. 2, pp. 1342-1347). IEEE.

Kreter, S., Schutt, A., & Stuckey, P. J. (2017). Using constraint programming for solving RCPSP/max-cal. *Constraints*, *22*(3), 432-462.

Kuo, Y., & Liu, C. C. (2017). Operator assignment in a labor-intensive manufacturing cell considering inter-cell manpower transfer. *Computers & Industrial Engineering*, *110*, 83-91.

Kuo, Y., & Yang, T. (2007). Optimization of mixed-skill multi-line operator allocation problem. *Computers & Industrial Engineering*, *53*(3), 386-393.

Lan, C. H., & Lan, T. S. (2005). A combinatorial manufacturing resource planning model for long-term CNC machining industry. *The International Journal of Advanced Manufacturing Technology*, *26*(9), 1157-1162.

MacCarthy, B. L., Wilson, J. R., & Crawford, S. (2001). Human performance in industrial scheduling: a framework for understanding. *Human Factors and Ergonomics in Manufacturing & Service Industries*, *11*(4), 299-320.

Mahdavi, I., Aalaei, A., Paydar, M. M., & Solimanpur, M. (2010). Designing a mathematical model for dynamic cellular manufacturing systems considering production planning and worker assignment. *Computers & Mathematics with Applications*, *60*(4), 1014-1025.

Mathlouthi, I., Gendreau, M., & Potvin, J. Y. (2018). Mixed integer linear programming for a multi-attribute technician routing and scheduling problem. *INFOR: Information Systems and Operational Research*, *56*(1), 33-49.

Nembhard, D. A. (2001). Heuristic approach for assigning workers to tasks based on individual learning rates. *International Journal of Production Research*, *39*(9), 1955-1968.

Norman, B. A., Tharmmaphornphilas, W., Needy, K. L., Bidanda, B., & Warner, R. C. (2002). Worker assignment in cellular manufacturing considering technical and human skills. *International Journal of Production Research*, *40*(6), 1479-1492.

Novas, J. M., & Henning, G. P. (2014). Integrated scheduling of resource-constrained flexible manufacturing systems using constraint programming. *Expert Systems with Applications*, *41*(5), 2286-2299.

Pabla, B. S., & Adithan, M. (1994). *CNC machines*. New Age International.

Pentico, D. W. (2007). Assignment problems: A golden anniversary survey. *European Journal of Operational Research*, *176*(2), 774-793.

Polo-Mejía, O., Artigues, C., Lopez, P., & Basini, V. (2020). Mixed-integer/linear and constraint programming approaches for activity scheduling in a nuclear research facility. *International Journal of Production Research* , *58* (23), 7149-7166.

Reddy, M. S., Ratnam, C., Rajyalakshmi, G., & Manupati, V. K. (2018). An effective hybrid multi-objective evolutionary algorithm for solving real time event in flexible job shop scheduling problem. *Measurement*, *114*, 78-90.

Rodammer, F. A., & White, K. P. (1988). A recent survey of production scheduling. *IEEE transactions on systems, man, and cybernetics*, *18*(6), 841-851.

Rodriguez, J. (2007). A constraint programming model for real-time train scheduling at junctions. *Transportation Research Part B: Methodological*, *41*(2), 231-245.

Rossi, F., Van Beek, P., & Walsh, T. (Eds.). (2006). *Handbook of constraint programming*. Elsevier.

Santos, F., Fukasawa, R., & Ricardez-Sandoval, L. (2021). An integrated machine scheduling and personnel allocation problem for large-scale industrial facilities using a rolling horizon framework. *Optimization and Engineering*, *22*(4), 2603-2626.

Sheikhalishahi, M., Azadeh, A., Pintelon, L., Chemweno, P., & Ghaderi, S. F. (2016). Maintenance scheduling optimization in a multiple production line considering human error. *Human Factors and Ergonomics in Manufacturing & Service Industries*, *26*(6), 655-666.

Süer, G. A. (1996). Optimal operator assignment and cell loading in labor-intensive manufacturing cells. *Computers & Industrial Engineering*, *31*(1-2), 155-158.

Süer, G. A., & Alhawari, O. (2013). Operator assignment decisions in a highly dynamic cellular environment. In *Industrial Engineering: Concepts, Methodologies, Tools, and Applications* (pp. 1135-1152). IGI Global.

Süer, G. A., & Tummaluri, R. R. (2008). Multi-period operator assignment considering skills, learning and forgetting in labour-intensive cells. *International Journal of Production Research*, *46*(2), 469-493.

Süer, G. A., Kamat, K., Mese, E., & Huang, J. (2013). Minimizing total tardiness subject to manpower restriction in labor-intensive manufacturing cells. *Mathematical and Computer Modelling*, *57*(3-4), 741-753.

Tortora, A. M., Di Pasquale, V., Franciosi, C., Miranda, S., & Iannone, R. (2021). The Role of Maintenance Operator in Industrial Manufacturing Systems: Research Topics and Trends. *Applied Sciences*, *11*(7), 3193.

Turkcan, A., Akturk, M. S., & Storer, R. H. (2003). Non-identical parallel CNC machine scheduling. *International Journal of Production Research*, *41*(10), 2143-2168.

Walsh, T. (2002, July). Stochastic constraint programming. In *ECAI* (Vol. 2, pp. 111-115).

Yang, I. T., & Chou, J. S. (2011). Multi-objective optimization for manpower assignment in consulting engineering firms. *Applied Soft Computing*, *11*(1), 1183-1190.

Yeh, W. C., Lai, P. J., Lee, W. C., & Chuang, M. C. (2014). Parallel-machine scheduling to minimize makespan with fuzzy processing times and learning effects. *Information Sciences*, *269*, 142-158.

Young, K. D., Feydy, T., & Schutt, A. (2017, August). Constraint programming applied to the multi-skill project scheduling problem. In *International Conference on Principles and Practice of Constraint Programming* (pp. 308-317). Springer, Cham.

Yun, Y. S., & Gen, M. (2002). Advanced scheduling problem using constraint programming techniques in SCM environment. *Computers & Industrial Engineering*, *43*(1-2), 213-229.

Zaman, T., Paul, S. K., & Azeem, A. (2012). Sustainable operator assignment in an assembly line using genetic algorithm. *International Journal of Production Research*, *50*(18), 5077-5084.

Zeballos, L. J. (2010). A constraint programming approach to tool allocation and production scheduling in flexible manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, *26*(6), 725-743.

Zhang, S., & Wang, S. (2018). Flexible assembly job-shop scheduling with sequence-dependent setup times and part sharing in a dynamic environment: Constraint programming model, mixed-integer programming model, and dispatching rules. *IEEE Transactions on Engineering Management*, *65*(3), 487-504.

Zhu, H., Chen, M., Zhang, Z., & Tang, D. (2019). An adaptive real-time scheduling method for flexible job shop scheduling problem with combined processing constraint. *IEEE Access*, *7*, 125113-1251