Filipe Ribeiro Saudade e Silva

# SCHEDULE OF MOLDS MANUFACTURING PROCESSES THROUGH PROCESS MINING

**Dissertation in the context of the Master in Informatics Engineering, specialization in Software Engineering, advised by Prof. Mário Alberto Zenha-Rela and by Prof. Rui Pedro Charters Lopes Rijo, and co-advised by Prof. Ricardo Filipe Gonçalves Martinho and Prof. Carlos Fernando de Almeida Grilo, and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.**

September of 2023

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE
COIMBRA

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Filipe Ribeiro Saudade e Silva

# Calendarização de processos de fabrico de moldes através de Process Mining

Dissertação no âmbito do Mestrado em Engenharia Informática, especialização em Engenharia de Software, orientada pelo Prof. Mário Alberto Zenha-Rela e pelo Prof. Rui Pedro Charters Lopes Rijo, e coorientada pelo Prof. Ricardo Filipe Gonçalves Martinho e Prof. Carlos Fernando de Almeida Grilo, e apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro de 2023

This work is funded by the project Prom4Prod (grant number CENTRO-01-0247-FEDER-047242), co-financed by the European Regional Development Fund (FEDER), through Portugal 2020, and by the Competitiveness and Internationalization Operational Programme (COMPETE 2020).

# Acknowledgements

I would like to express my sincere gratitude to everyone who has contributed to the successful completion of my master's dissertation. This journey has been both challenging and rewarding, and I could not have accomplished it without the support and assistance of various individuals.

First and foremost, I am thankful to my academic advisors and co-advisors, Professor Mário Alberto Zenha-Rela, Professor Rui Pedro Charters Lopes Rijo, Professor Ricardo Filipes Gonçalves Martinho, Professor Carlos Fernando de Almeida Grilo for their guidance, insights, and constant encouragement throughout the research process. Their knowledge was crucial in determining the direction of this work.

I am indebted to my family, especially my mother Elsa Ribeiro, for their unconditional love, support, and permanent belief in my abilities. Your constant encouragement has served as my foundation and motivation throughout my academic career.

Lastly, I extend my heartfelt appreciation to Mariana Subtil for her love and support, and my friends, who have been a source of motivation and comradeship. Your discussions and feedback have been immensely helpful in refining my ideas and approaches.

This dissertation would not have been possible without the collective efforts and support of all those mentioned and the countless others who have contributed in their own ways. Thank you for being an integral part of this significant accomplishment.

# Abstract

In the mold industry, being a highly competitive sector, organizations are constantly seeking ways to optimize their business processes in order to gain a competitive edge. This has led to a push for the integration of Industry 4.0 technologies in order to gain access to advanced management tools. Consequently, the amount of manufacturing process data significantly increased, allowing for more accurate and real-time process analyses. This is also the case of the molds industry, where nowadays, we continue to observe that organizations face numerous challenges, including suboptimal resource management, uncertainties in providing accurate delivery dates to clients, and a lack of visibility into ongoing processes on the production floors. These issues can prevent the ability to track the current status of production for specific molds or parts of a mold, leading to inefficiencies and potential customer dissatisfaction.

To address these challenges, a study was carried out in which we employed Process Mining (PM) algorithms to analyze real-world data on production processes within the mold industry. These data were used as metrics for a Genetic Algorithm (GA), which was employed to optimize and determine the most feasible schedules for the production floors.

By integrating PM and GA, our approach seeks to increase scheduling efficiency, resource utilization, and visibility into ongoing production processes.

Through this approach, organizations in the mold manufacturing industry can achieve greater process control, reduce delays, and gain the ability to provide more accurate delivery dates to clients, thereby enhancing their overall market competitiveness.

# Keywords

Process Mining, Petri Nets, Genetic Algorithm, Business Process Discovery, Mold Manufacturing, Mold Industry.

## Resumo

Na indústria dos moldes, sendo este um sector altamente competitivo, as organizações procuram constantemente formas de otimizar os seus processos empresariais de modo a obterem uma vantagem competitiva. Isto leva a que exista um impulso para a integração de tecnologias da Indústria 4.0, com o intuito de obter acesso a ferramentas de gestão mais avançadas. Consequentemente, a quantidade de dados resultantes dos processos de fabrico aumentou significativamente, permitindo análises mais precisas aos processos e em tempo real. Este é também o caso da indústria de moldes, onde, atualmente, continuamos a observar que as organizações enfrentam inúmeros desafios, incluindo uma gestão de recursos insuficiente, incertezas no fornecimento de datas de entrega suficientemente precisas aos clientes e uma falta de visualização dos processos em curso no chão de fábrica. Estes problemas podem impedir a capacidade de acompanhar o estado atual da produção de moldes específicos ou peças de moldes específicas, resultando em ineficiências e na potencial insatisfação dos clientes.

Para responder a estes desafios, foi realizado um estudo em que utilizámos algoritmos de *Process Mining (PM)* para analisar dados reais sobre os processos de produção na indústria de moldes. Estes dados foram utilizados como métricas para um *Genetic Algorithm (GA)*, que foi aplicado para otimizar e determinar os escalonamentos mais viáveis para os pisos de produção.

Ao integrar PM e GA, a nossa abordagem visa aumentar a eficiência dos escalonamentos, a utilização de recursos e a visibilidade dos processos de produção em curso.

Através desta abordagem, as organizações presentes na indústria de fabrico de moldes podem obter um maior controlo sobre os processos, reduzindo os atrasos e obter a capacidade de fornecer datas de entrega mais precisas aos seus clientes, aumentando assim a sua competitividade no mercado em geral.

## Palavras-chave

*Process Mining*, *Petri Nets*, *Genetic Algorithm*, *Business Process Discovery*, Fabrico de Moldes, Indústria de Moldes.

# Contents

# Acronyms

**API** Application Programming Interface.

**BPM** Business Process Management.

**BPMN** Business Process Model and Notation.

**CRISP-DM** Cross Industry Standard Process for Data Mining.

**CRM** Customer Relationship Management.

**DFG** Directly-Follows Graph.

**ERP** Entreprise Resource Planning.

**FPSP-MM** Fuzzy Production Scheduling Problem considering Mould Maintenance.

**GA** Genetic Algorithm.

**IoT** Internet of Things.

**JSP** Job-shop Scheduling Problem.

**KPI** Key Performance Indicator.

**PM** Process Mining.

**PS-MM** Production Scheduling Problem with Mold Maintenance.

**RCMPSP** Resource-Constrained Multiple Project Scheduling Problem.

**SA** Simulated Annealing.

**SaaS** Software-as-a-Service.

**SI** Swarm Intelligence.

**TCPN** Time Colored Petri net.

**TLPSO-VNS** Three-level Particle Swarm Optimization with Variable Neighborhood Search Algorithm.

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The rise of Industry 4.0 – the integration of Internet of Things (IoT), cloud computing, data processing and analytics into the factories' infrastructure, the automatic collection, transformation, and processing of data from manufacturing floors have become feasible. Given this transformative potential, it is wise to investigate these events to retrieve feedback regarding the existent processes. Buzzwords like *business activity monitoring* and *business process intelligence* imply that organizations and software developers want solutions that can extract insight from so-called *event logs* [van der Aalst and Weijters, 2004], which include, for each record, a certain product or service identifier, a timestamp, an activity and the resources needed to perform it. These concede the application of Process Mining (PM) algorithms to discover, compare and enhance manufacturing processes. PM is an analytical discipline that provides a set of tools that allow insights for the optimization of business processes in a given environment. These tools are composed by algorithms that transpose the events captured in the event logs and pinpoint their relation with the business logic and how they were executed.

In this work, we divert our attention to the mold industry, which is characterized by having an *ad hoc* manufacture, meaning that no two molds are produced the same way, due to not only being different from each other, but also for having a singular production context including, for instance, resources' availability at certain moment, time constraints and the overall activity in the factory, resulting in unstructured business processes, also known in literature as *Spaghetti processes* [van der Aalst, 2011].

This work aims to improve the scheduling of activities within the mold industry's manufacturing processes, a notoriously unpredictable industry, by employing process mining algorithms to uncover a more accurate representation of the business processes performed on the shop floor. By doing so, we want to create schedules that are more aligned with industry needs and increase overall production efficiency.

One of the most difficult aspects of this work is to effectively organize, clean, and standardize the data collected from the mold industry's manufacturing floors. This enables the uncovering of meaningful insights and patterns in the process, which in turn allows for more optimal resource planning. The process of data preparation can be time-consuming and requires a detailed understanding of the data, including the meaning and relationships of the different fields in the database. Despite the challenges, this step is essential to the success of our research, as it establishes the basis for accurate and effective scheduling of activities in the mold industry.

## 1.1 Context

This work is part of a wider scoped project named Prom4Prod, which is co-funded by the *Portugal 2020* program. The goal of this wider project is to create a decision support, planning, and production management system for engineer-to-order job-shop production industries. This work is validated for the case of the plastics injection molds industry, using real data obtained from historical process execution of the factory floor.

The Prom4Prod project comprises the research, design, and validation of several industry-relevant components, innovative management approaches, and the comprehensive development of a decision support system. The research primarily involves the application of Process Mining (PM) techniques to analyze engineer-to-order production systems. By using PM, the system provides users with valuable data to support in production, resource management, and other crucial decision-making processes.

Moreover, the developed system undergoes thorough validation using real production data acquired from the mold company within the project's consortium. The design and validation of a decision support system are based on real processes, including the use of heuristics and analysis of similar situations. This approach aims to enhance the system's ability to offer valuable insights and support informed decision-making for optimizing production processes within the industry.

In this work, we primarily focus on addressing objectives related to the scheduling of molds manufacturing processes, and their validation using real data. Our aim is to develop a module to be integrated into the Prom4Prod infrastructure to capture the composition of previously built molds using PM in order to generate schedules for the production of individual parts of the molds.

## 1.2 Problem and Motivation

The molds industry is a highly competitive sector, and as a result, it is subjected to a significant amount of pressure on costs and production times. This pressure drives the sector to become more efficient in order to have the best decision-making mechanisms, which in turn affects their business processes with the goal of reducing inefficiencies.

The challenge in mold manufacturing is that each mold is a one-of-a-kind, made-to-order production, undergoing a series of unique operations due to their own particular characteristics, which leads to frequently occurring adjustments in their planning [Mourtzis et al., 2014]. Therefore, each mold is also completely unique in terms of the manufacturing business processes that are carried out. As a result, planning decisions and their changes are always dependent on a triple: the mold's specific planning, the production status of the several existing molds, and the processes to be concretely performed on the production line.

As witnessed by members involved in the Prom4Prod project, in the real-world, manufacturers attempt to incorporate each mold into their manufacturing processes wherever feasible, taking into account primarily the resources available at the moment, usually in a first come first-served fashion, while still recognizing the value of the customers who have placed orders. This causes a variety of issues, including deviations in the molds produced, bottlenecks within the manufacturing processes, and non-conformities to their ideal process models.

Currently, the factory floor planning is carried out manually, taking a top-down approach. Unfortunately, this method is prone to errors due to its lack of granularity. Given the existing circumstances, this manual approach remains their only known method. This process consumes a significant amount of time, resulting in infrequent scheduling. Consequently, the current situation perpetuates a cycle where poor scheduling gives rise to *ad hoc* processes, further worsening the accuracy of scheduling.

## 1.3 Objectives

The mold industry is characterized by its ad hoc techniques and having a high complexity, which in turn leads to unstructured processes as it operates on a make-to-order basis. However, by structuring, cleaning, and filtering the area of the mold and by its attributes (e.g., number of cavities, number of parts and specific part types) we believe that it is possible to achieve a semi-structured or even structured processes that can be automated.

In this work, the Prom4Prod platform is enhanced by incorporating a new feature. The primary purpose of this addition is to automate the scheduling of multiple manufacturing processes in mold production. To achieve this, the system considers various factors such as the context of each process, the resource occupancy, and the historical workflow data gathered through the implementation of Process Mining (PM) algorithms. The integration of these elements aim to optimize the overall manufacturing workflow and improve the efficiency of mold manufacturing processes.

With this approach, production engineers are provided with real-time support in making decisions, including suggested activities for the molds being produced and the allocation of appropriate resources. This approach proves to be more flexible and adaptable than the commonly used manual scheduling method in this industry. Since the system can adapt to the production floor, which makes it a better way to schedule work.

Furthermore, by using this approach, production engineers can optimize the allocation of resources and improve the overall workflow efficiency in mold manufacturing. The system's ability to adapt to changing conditions allows for better resource management and timely decision-making. This, in turn, contributes to a more streamlined and effective production process, leading to improved productivity and customer satisfaction.

## 1.4 Document Structure

The remainder of this report is structured as follows:

- **Background:** Addresses the key concepts, problem description, and previous work developed in the research areas related to the project;
- **State of the Art:** Describes the state of the art by providing an overview of prior research on scheduling processes and the use of PM techniques within the context of the mold industry.
- **Research and Development Methodologies:** A description of the methodologies used in the research and development process is provided, including their procedures and implementation;
- **Development:** The solution is explained by first introducing an explanation of the project requirements. Subsequently, it introduces the chosen architecture and the steps involved, along with a description of the developed genetic algorithm.
- **Results and Discussion:** The results of validation, comparative, and performance tests are presented, followed by a discussion of the advantages and limitations of the work;
- **Conclusion and Future Work:** A final balance of this work's results and achievements, in relation with the previously established objectives. Furthermore, potential approaches for future research and development are outlined.

# Chapter 2

# Background

This chapter provides an overview of the project's key aspects. These include the mold industry's business context, the application of Process Mining (PM), the differentiation between structured and unstructured processes, and the challenges associated with their activities scheduling.

## 2.1 Mold Industry

The mold industry holds a significant position within the manufacturing sector, as molds are indispensable tools used in a wide range of manufacturing processes. Molds serve a crucial role in transforming basic materials, such as plastic, metal, or glass, into the precise shapes required for mass production. Molds are widely used in industries such as automotive, electronics, packaging, and consumer goods, allowing the manufacture of products with complex geometry with notable precision and repeatability, thereby ensuring consistent product quality.

Our work focuses on the domain of plastic injection molding. This specific focus originates from the availability of data acquired from the Prom4Prod initiative, which includes organizations of the plastics injection molds industry. Injection molds are used for injecting molten materials, such as plastic or metal, into a mold cavity to create a desired product shape. These molds consist of two halves, the core and one or more cavities, which are designed to create the desired product geometry.

The Portuguese mold industry has established a standard for classifications and designations for various parts of molds used in the production of plastic materials [Silva, 2015]. To assist the learning and training in this industry, a dedicated website has been created, providing an introduction to mold manufacturing[1]. This training manual not only covers the classification of mold components, but also the typical machines found on factory floors and their functions. By getting familiar with these vital parts and machines, we intend to establish a contextual understanding of the provided data.

[1]`http://formacao.training.pt/`

Within our work, we have directed our attention to two crucial components: bushings and cavities. Cavities play a vital role in mold production, as they shape and form plastic materials into desired shapes, influencing the final product. They are a distinct entity within the mold structure. On the other hand, bushings are separate components that guide the movement of mold components, ensuring precise alignment and smooth operation. They facilitate material flow and contribute to the mold's proper functioning during production.

This focus is primarily driven by the abundance of data available, as these two parts are the most frequently produced for any mold. In the standardized classification, cavities are assigned the code 100, while bushings are assigned the code 200. Any other components not explicitly defined in the standard are classified under the group of parts labeled as 900.

### 2.1.1 Machinery

Throughout the manufacturing process of a part, a wide range of machines are available to fulfill various operations. These machines play a vital role in shaping and fine-tuning the parts to meet specific requirements. Based on the available data, there are several common groups of machines that are frequently found on factory floors within the mold industry:

- **CNC Machines:** Machines controlled by pre-programmed code, such as grinders, lathes, and turning mills, all of which are used to cut, shape, and create different parts out of a steel workpiece;
- **Milling Machines:** Machines designed to cut pieces of the most varied geometric shapes. Usually use rotating tools (e.g., milling cutters or drills), which cuts the material fixed in a clamping device;
- **Grinding Machines:** Designed to remove excess material from workpieces, creating smooth surfaces with precise dimensions;
- **Electrical Discharge Machines:** Machines which use controlled electrical discharges to shape and refine materials with more precision, making them well-suited for complex cuts;
- **Wire Electrical Discharge Machines:** Machines which employ a thin metal wire as an electrode, which is guided through the workpiece, useful for materials that are difficult to machine using conventional methods.

Some additional processes in the mold industry involve a more manual approach, where workers rely on a workbench to perform tasks such as polishing, fine-tuning, and assembly of the individual mold components. These manual operations require human expertise and attention to detail to ensure the desired quality and finish of the pieces.

### 2.1.2 Production Process of a Mold

The production process of a mold typically encompasses six primary stages:

1. **Budget Agreement:** This initial stage involves reaching an agreement on the budget for the mold production, ensuring that cost considerations are aligned with the customer's requirements;

2. **Preliminary Design:** In this stage, a preliminary design is developed, taking into account factors such as desired specifications and functionality. This phase serves as the foundation for the subsequent development process;

3. **Development:** The development stage incorporates various activities, including rheological studies to analyze the flow characteristics of the materials, planning the desired mold structure, and identifying specific parts required for the mold assembly;

4. **Production:** The production stage involves the actual manufacturing of each mold component and the subsequent assembly of the mold itself. Skilled technicians and specialized machinery are employed to ensure precision and quality during this phase;

5. **Testing:** Once the mold is completed, testing is conducted to assess its quality (i.e., its performance, functionality, and durability). These tests help identifying any potential issues or areas for improvement before the final product is delivered to the customer;

6. **Expedition:** In the final stage, the mold is packaged and shipped to the customer, ready for use in their mold injection processes.

### 2.1.3 The Scheduling Problem

In the context of our case study involving a consortium of companies in the mold industry, it has become evident, that the scheduling of mold manufacturing processes on the factory floor is inefficient and faces significant labor costs.

This inefficiency results from the need for an expert with both technical expertise and an in-depth knowledge of the business dynamics. This person must be capable of determining which parts should be prioritized, identifying appropriate resources, and overseeing ongoing processes while considering factors such as customer context and prioritization.

Furthermore, the scheduling process is currently handled manually, undertaken from a broad perspective on a weekly basis. Unfortunately, this method is prone to errors due to its lack of granularity, and given the current circumstances, this manual method remains the only approach they know and use. This process is costly in terms of time, resulting in less frequent scheduling.

Consequently, the current situation resembles a cycle where poor scheduling creates room for *ad hoc* processes, which in turn leads to even less accurate scheduling. The ideal solution would be to establish more frequent and accurate schedules. Thus, the primary objective of this work is to automate the scheduling process.

This automation also aims to relieve the scheduler from this duty, allowing them to focus on other responsibilities, but also achieving more complex and optimal schedules. Among the primary challenges within the scheduling process is the integration of quality control into the manufacturing workflow. This integration renders significant process variability, as activities may need to be repeated until the component aligns with the quality standards set by the operator.

In essence, the mold industry faces complex scheduling challenges due to the need for expertise consolidation, coupled with the variability in manufacturing processes, and the fundamental goal of acquiring a more refined scheduling technique. As a vital component of the manufacturing sector, the mold industry enables the production of precise molds for large-scale manufacturing of diverse products.

Through our focused study on plastic mold injection, we have gathered data from the Prom4Prod project to gain insights into the classification and designation of mold components, placing particular emphasis on bushings and cavities. This study has also involved an examination of commonly used machines and the overall existing process in mold manufacturing. By comprehending these critical components and their associated data, we have cultivated a contextual understanding of the mold manufacturing process.

## 2.2 Process Mining

Over the past decade, PM has grown and became an independent field of research. In conventional Business Process Management (BPM), workshops and interviews are used to provide an idealized representation of a business process (a process model). The PM approach, on the other hand, employs existing data from corporate information systems to expose the actual business processes in an automated fashion while also incorporating techniques for their discovery, monitoring, and enhancement [Corallo et al., 2020; Lamghari, 2022].

Unlike Business Intelligence tools, PM digs inside historical data by using *event logs* to check for casual dependencies, bottlenecks and deviations. Event logs include important details like the time and date of an event, its actual description, and the resources that were used. These can originate from all kinds of systems, ranging from embedded systems in automobiles or machines to Enterprise Information Systems from organizations such as hospitals or factories.

Process Mining is a set of analytical techniques that uses the data stored in the event logs. These approaches assume that it is possible to extract a sequence such that each event corresponds to an activity (i.e., one of the steps in a business process that specifies a particular action to be carried out) and is related to a particular case (i.e., an instance of the process representational model). This makes it possible to provide organizations with more information about what is actually happening.

In the case of the molds industry, PM helps decision makers understand the processes on the factory floors not only by displaying them and their deviations, but also by enabling the checking of conformance between expected and real workflows. This allows for the identification of not only existing bottlenecks and problems, but also potential ones that the factory floors may face [Trzcionkowska and Brzychczy, 2018].

We can consider that there are three basic types of process mining techniques [van der Aalst, 2016]:

- **Process Discovery:** Usage of event data to develop realistic process models. These can be expanded to demonstrate bottlenecks and outlier behavior;
- **Conformance Checking:** Check whether the observed processes correspond to the modeled (ideal) processes and vice-versa;
- **Enhancement:** Extend or enhance an existing process model by integrating information from the actual process event log, to improve the following executions of the process.

Focusing on **Process Discovery**, there are multiple PM algorithms integrated into this topic, the first ever developed being the $\alpha$-algorithm (also known as alpha miner).

The $\alpha$-algorithm extracts various relations between the activities occurring in the event log. These relations are used to generate a Petri net that reflects the process. Although the $\alpha$-algorithm is fairly simple, it provides a good introduction to the topic, since it represented the basis for the development of other algorithms (e.g., heuristics miner) [van der Aalst et al., 2004].

Other popular process discovery algorithms include the heuristics miner and the inductive miner.

The heuristics miner identifies the underlying process flow by applying a set of heuristics, or rules, to the event log data. It can handle event logs with a high level of noise and incompleteness, making it a popular option for real-world applications where data may not be perfect [Weijters et al., 2006].

The inductive miner is another process discovery algorithm used to extract process models from event logs. It identifies patterns in the event log data and discovers the underlying process flow using statistical analysis and machine learning techniques. The inductive miner can deal with large and complex event logs, and it can detect both control-flow and data-flow dependencies. However, it is sensitive to noise and thus, it may need a significant amount of data to produce accurate results [van der Aalst, 2016].

### 2.2.1 Key Software Tools

There are several commonly used software tools that allow the usage of PM algorithms.

ProM (The Process Mining Toolkit)[2] is among the most widely used tools for PM. ProM is a Java-based, open-source application that supports a number of PM methods and approaches. Because it is open-source, several plugins have been integrated into it, allowing it to handle a wide range of input and output formats.

Disco[3] is a popular commercial process mining software tool, famous for its simplicity and fast results. It offers a range of features for process discovery, process conformance, and process improvement.

Celonis[4] is another commercial process mining tool that specializes in analyzing data from various systems, such as Entreprise Resource Planning (ERP) and Customer Relationship Management (CRM) systems. It can discover bottlenecks, inefficiencies, and improvement opportunities in real-time by having the ability to extract data from a variety of databases types. Which makes it a versatile tool that can be used in different industries.

Another option is PM4Py (Process Mining for Python)[5], which is an open-source library for Python. It provides a wide range of options, covering all the three basic types of PM techniques and offers the advantage of customization and the ability to create plugins, similar to ProM.

Overall, each of these four most well-known PM tools have their own advantages and disadvantages. Some are commercial products, while others are limited by the programming language in which they are built in. Nevertheless, they all provide users with a range of capabilities for analyzing business processes and improving their efficiency and effectiveness.

### 2.2.2 Process Representation

In the field of process mining, there is a range of methods to depict the workflows extracted from the analyzed processes. These graphical representations are effective instruments for comprehending the elaborate flux of activities, decisions, and dependencies within a process. Some of the common ways to represent these workflows include Business Process Model and Notations (BPMNs), Directly-Follows Graphs (DFGs), and Petri nets.

BPMN is a widely-used standard for modeling business processes. It uses simple symbols and graphical components to represent different types of tasks, events, gateways, and flows. BPMN diagrams provide a clear overview of process activities and their sequences, which makes them an excellent choice for depicting

[2]`https://promtools.org`

[3]`https://fluxicon.com`

[4]`https://celonis.com`

[5]`https://pm4py.fit.fraunhofer.de`

process structures [White, 2004].

DFGs provide an informative yet simplified view of process sequences. They prioritize portraying direct relationships between activities, highlighting their frequency and order of incidence. DFGs are especially helpful for identifying common paths and obstacles within a process.

In this study, we focus on the representation of process workflows using Petri nets. Petri nets are formal models that excel at representing processes' complex relationships, parallelism, and conditional behavior. They consist of places, transitions, tokens, and arcs that collectively represent the flow of activities and decisions. Petri nets offer a significant advantage over other techniques of representation as they can not only provide a clear visualization of the process structure, particularly in detecting simultaneous execution of activities, but also have the possibility to include performance and frequency metrics.

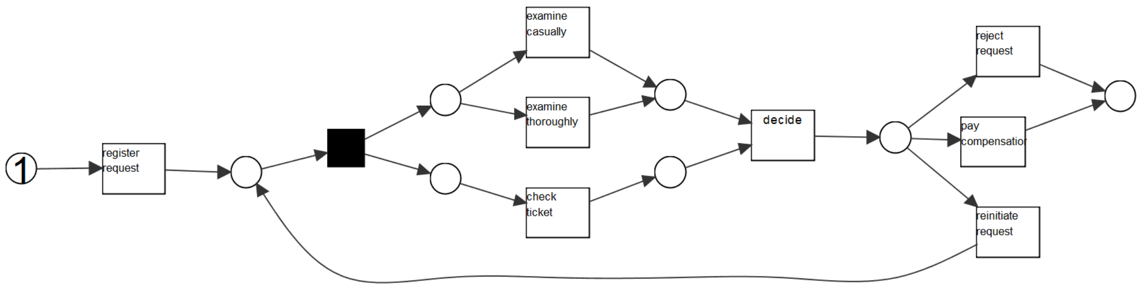


Figure 2.1: Example of a Petri net adapted from [Fraunhofer Institute, 2021].

A simple Petri net generated from the synthetic event log "running-example.csv" from the PM4Py framework is displayed in Figure 2.1.

Petri nets are structured diagrams consisting of places (depicted as circles) and transitions (represented as rectangles) interconnected by arcs (depicted as lines forming the relations). The diagram begins with a start place and concludes with an end place, forming the foundation for illustrating process workflow.

Each place in this diagram is only connected to transitions, while transitions are only connected to places. This structural arrangement assures the logical consistency of the representation.

Silent transitions, denoted by the black rectangles, play a unique role in the Petri net. Unlike standard transitions linked to events in the event log, silent transitions are absent of direct connections to the event log. Instead, they serve as vital routing mechanisms, allowing the flow of tokens (i.e., symbols representing the process of work between places and transitions).

Now, we shall analyze the particular features highlighted by this Petri net. The initial task executed is the "register request". Following this, the parallel execution of two workflows emerges. This parallel execution is visualized through the silent transition that diverges into two branches. Notice that the place preceding "examine casually" and "examine thoroughly" contains two connections, denoting an "OR" relationship. This indicates that only one of these relations can be "active" at any given time within the process, reflecting a workflow decision point.

Overall, this visualization contributes in comprehending the dynamics of the underlying process and offers valuable insights into the execution of activities and their relationships.

## 2.3 Structured and Unstructured Processes

The use of PM techniques in various fields revealed a distinction between two kinds of existing processes: structured processes, also known as *Lasagna Processes*, and unstructured processes, also known as *Spaghetti Processes*. We may also distinguish an intermediate form, known as "semi-structured" processes [van der Aalst, 2011; van der Aalst and Gunther, 2007].

*Lasagna Processes* are predictable and well-defined (like a lasagna, for which we can immediately identify each layer, as well as its structure as a whole), allowing a major portion of them to be automated.

In Figure 2.1 presented in the previous section, we can observe an example of a *Lasagna Process*, drawn from a small sample of event logs.

In "semi-structured" processes, it is possible to recognize its activities, yet certain activities may deviate from the main sequence based on the observed situation, necessitating human review for their interpretation.

Finally, the *Spaghetti Processes* are highly difficult to define since most of their cases have their own deviations, making it impossible to scope the primary sequence in them. These are human-driven processes and, because there are usually only suggested guidelines, the processes are affected by their intuition, experience, or “trial-and-error” methods, making it difficult to achieve a meaningful analysis. However, by carefully filtering the data and focusing on a particular portion of the process, it is possible to uncover hidden patterns that hold the potential to enhance the overall process.

An example of a *Spaghetti Process* generated using real data can be seen in Figure 2.2.

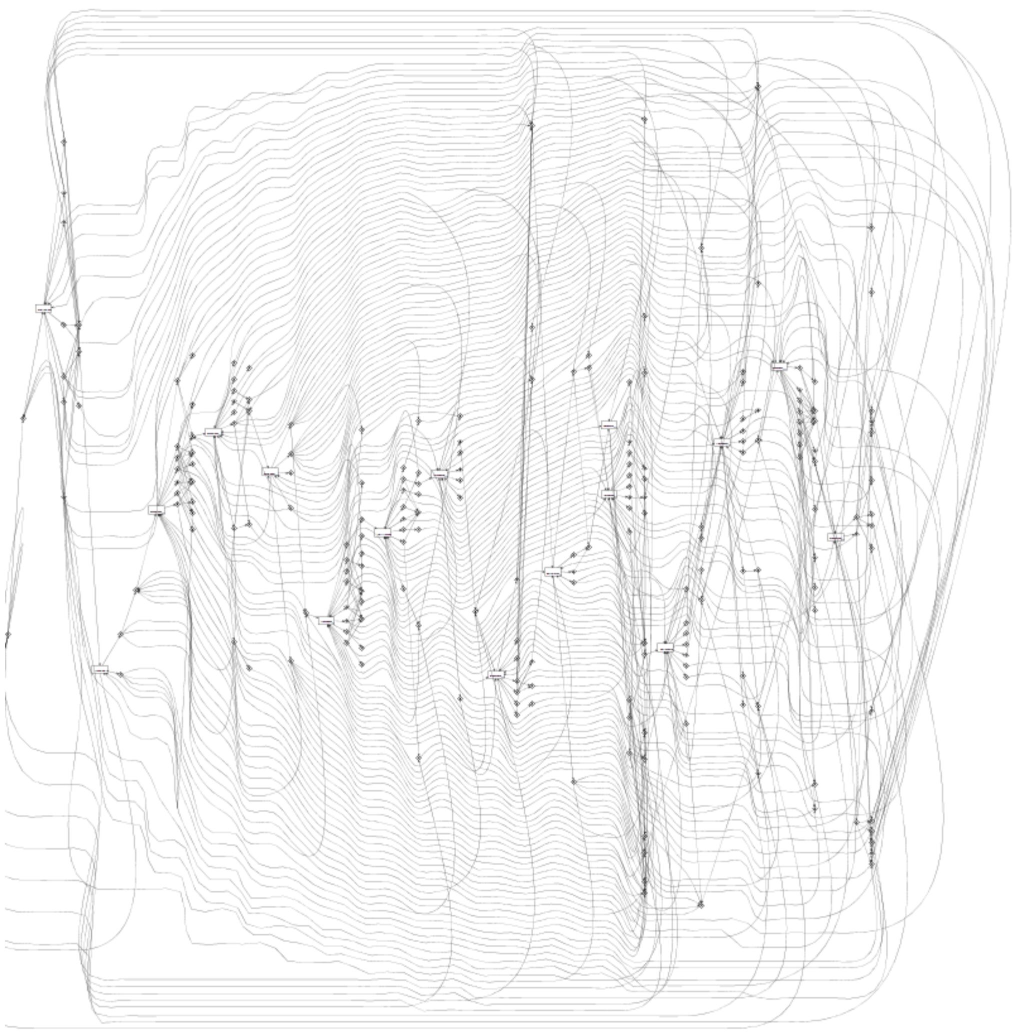


Figure 2.2: Example of a Spaghetti Process.

Even though *Spaghetti Processes* are more difficult to analyze than *Lasagna Processes*, they still provide valuable insights and have significant room for enhancement.

Event logs do not accurately portray the entire environment, e.g. they can only transmit example behavior [van der Aalst and Gunther, 2007], since they record just a fraction of the possible paths. Therefore, process discovery becomes difficult since it is impossible to forecast alternative patterns. As *Wil van der Aalst*, the father of process mining, stated [van der Aalst, 2011]:

> *"The fact that something does not happen in an event log does not mean that it cannot happen."*

In case of the molds industry, a mold's production may have certain features similar to other molds, but the method by which each kind of mold is built can be entirely different. In other words, while dealing with mold industry data without any form of filtering or focusing on specific sections of the mold-making process, we are presented with *Spaghetti Processes*.

## 2.4 Job-shop Scheduling Problem

Various industries, including manufacturing, production planning, computer design, logistics, and communication, experience machine scheduling issues [Cheng et al., 1996].

The Job-shop Scheduling Problem (JSP) is considered one of the existing intractable numerical problems, also known as NP-hard problems, and is one of the most important and challenging in the field of operations management [Manne, 1960].

The generic JSP problem is defined by a set of jobs and a set of machines. Each job consists of a group of operations that must be carried out in a certain time frame on a particular machine without interruption. Each machine is only capable of doing one operation at a time. A schedule is the assignment of operations to machines at a predetermined time period. The purpose of this challenge is to determine the least amount of time required to schedule all existing jobs.

As one of the most difficult combinatorial optimization problems, it features a vast search space of possible solutions, which makes the use of heuristic procedures appealing [Zhang et al., 2017]. The adoption of stochastic search methods such as Tabu Search, Simulated Annealing (SA), Genetic Algorithm (GA) and Swarm Intelligence algorithms, seem to have a high success rate in identifying good-fitting solutions.

Tabu search employs local search techniques similar to those used in mathematical optimization to identify a probable solution and explore the neighborhood for its optimization [Glover, 1986].

SA is a stochastic technique for approximating the global optimum of a given function. In the case of JSP, this function consists of the solution search space and its fitness [Brabazon et al., 2016].

GA is based on natural selection and consists of using adaptive and global stochastic search algorithms in a space of candidate solutions while also being directed by an objective function in order to find the best fitting solution [Brabazon et al., 2016]. To achieve this, a GA exhibits four distinctive characteristics:

1. **Initialization of a Population:** The process begins with the creation of an initial population, each representing a potential solution to the problem. Typically, these individuals are generated through a random or heuristic process, resulting in a diverse set of candidate solutions;

2. **Selection of Individuals as Parents:** A subset of the current population is chosen to serve as parents for the next generation during the selection phase. The selection procedure gets influenced by each individual's fitness, as determined by the objective function. Individuals with higher fitness levels are more likely to be chosen, similar to the concept of "survival of the fittest" [Brabazon et al., 2016].

3. **Genetic Operators:** Responsible for creating new candidate solutions from existing ones. These operators mimic biological processes such as crossover

and mutation. In the crossover operation, genetic material from two parent solutions is combined to produce one or more offspring who might acquire beneficial traits from their parents. Mutation introduces random changes to the individual's genes, promoting the exploration of novel solution spaces and enabling the emergence of novel traits.

4. **Replacement of the Population:** The new generation of individuals, consisting of parents, children resulting from crossover, and individuals subjected to mutation, replaces some or all of the individuals in the current population. This replacement procedure assures the constant evolution and adaptation of the population over successive generations.

By iteratively applying these principles, Genetic Algorithms (GAs) explore the solution space, adapt to changing conditions, and seek the optimal solution based on the provided objective function.

Swarm intelligence employs the same ideas as a conventional GA but depends on decentralized, self-organizing individuals to discover solutions to the problems [Brabazon et al., 2016]. It consists of nature-inspired particle swarm optimization algorithms, with three main variations of swarm models:

- Flocking behavior of birds or the sociological behavior of a group of humans;
- Food foraging behavior;
- Social behavior of insects (as for example, ant colonies).

## 2.5 Conclusion

In conclusion, this chapter has provided foundations for our subsequent work. We have determined the crucial position of the mold industry in manufacturing, exposing both its significance and the challenges it faces. In addition, we introduced Process Mining as an effective technique for examining processes. Lastly, we described the complexities surrounding the Job-shop Scheduling Problem, a challenge that underpins much of our study.

Having established this foundational understanding, we proceed with a detailed exploration of the work that has been undertaken in the upcoming chapters, starting by delving into the state of the art in these research areas.

# Chapter 3

# State of the Art

The goal of this chapter is to describe the state of the art by providing an overview of prior research on scheduling processes and the use of PM techniques within the context of the mold industry. This includes a discussion of relevant work that has been conducted in this field. By reviewing this previous research, we can gain a better understanding of the context of the study and identify current gaps and acknowledged opportunities for further investigation.

## 3.1 Search Strategy

For the search on related work , we used *Google Scholar* and *Research Rabbit* as our preferred search engines to find the best fitting scientific articles. We used a set of keywords to search for information, such as *'"process mining" AND "mold manufacturing"'*, *'"scheduling" AND "mold industry"'*, *'"process mining" AND "scheduling"'* and *'"process mining" AND "scheduling" AND "mold industry"'*. These keywords allowed us to find a set of articles that fit completely or partially inside the context of our work.

Table 3.1 presents a classification of related articles by the main research areas surrounding this work. The ones that we found relevant to classify were: 1) Process Mining, if the work references the use of process mining techniques or outcomes resulting from those techniques; 2) Scheduling, whether it covers the scheduling of processes or simply JSP, which is split into four categories: Genetic Algorithm (GA), Simulated Annealing (SA), Swarm Intelligence (SI), or Others; 3) Mold Industry, if the study relates to the mold industry or mold manufacturing.

Table 3.1: Table of related work articles classified by the area of work.

| Article | Process Mining | Scheduling | | | | Mold Industry |
|---|---|---|---|---|---|---|
| | | GA | SA | SI | Other | |
| [Park and Choi, 2006] | | X | | | | |
| [Wu et al., 2009] | X | | | | X | X |
| [Liu et al., 2010] | | X | | | | X |
| [Tamilarasi and Anantha, 2010] | | X | X | | | |
| [Gomes et al., 2013] | | | | | X | X |
| [Caballero-Villalobos et al., 2013] | X | X | | | | X |
| [Mourtzis and Vlachou, 2018] | | | | | X | X |
| [Fu et al., 2019a] | | | | X | | X |
| [Fu et al., 2019b] | | | | X | | X |
| [Jong et al., 2020] | | X | | X | | X |
| [Lee et al., 2020] | | | | | X | X |
| [Choueiri and Santos, 2021] | X | X | | | | |

An integration of process planning and job shop scheduling using a genetic algorithm is explored by Park and Choi [2006]. The researchers adapted a GA previously developed by them to address the specific constraints encountered in process planning. The approach was tested using several benchmark cases, but it was not applied to real-world problems. Nonetheless, it proved to be a considerable performing approach where it could produce an optimal or improved solution for the JSP, meaning they could meet the due dates demanded by customers.

The use of Petri nets to solve a Resource-Constrained Multiple Project Scheduling Problem (RCMPSP) is approached by Wu et al. [2009]. Petri nets, being the product of Process Mining algorithms, have the benefit of being able to simulate dynamic, concurrent, and asynchronous activities compared to other modeling tools. This behavior helps in the solving of the RCMPSP, as we have a more realistic view of the processes. In this work, Time Colored Petri nets (TCPNs) are used, enabling a more compact and concise view on the process flow, since by introducing "colors" we are able to distinguish different entities and have additional logic inside the Petri nets. To test the suggested approach, the authors conducted a case study using simulated data to mimic the mold manufacturing business. The results show that using TCPNs is not only feasible but also effective. Additionally, it was found that the current algorithm is very CPU-inefficient and its optimization is one of the objectives in future research.

Another approach using Petri nets is described in Caballero-Villalobos et al. [2013] to compute the scheduling of complex manufacturing systems. This case study focuses on plastic injection molds and provides improvements to a prior research that employed Petri nets and Genetic Algorithms (GAs) to handle complex production scheduling problems. The key contributions of this study are the ability to model a broad range of manufacturing systems without modifying the net structure or the solutions representation (chromosome) and the optimization of the method used to generate the initial population of solutions. These have improved the quality of the results compared to the previous tested methods. Future research is expected to focus on reducing the consequences of redundant representations in an individual's chromosome through the use of other approaches in the discovery of sequence transitions in Petri nets.

Liu et al. [2010] explores the development of a GA with the objective of solving mold enterprise's JSP. They focused on mold production constraints with the goal of making the processing completion time of all work items the shortest possible, although the due date is not mentioned as a constraint in their processes. In their study, they made the GA account for the time required for each working procedure, the transit durations between each machine, and the preparation time for each machine to provide a schedule that is more representative of the actual process.

The work of Tamilarasi and Anantha [2010] reflects the possibility of using a hybridization of GA with SA to solve the NP-hard JSP. While not in the context of the mold's industry, it has an extensive practical description on how to conduct this hybridization. The experimental results obtained indicate that it is an optimization comparing the use of a GA alone, and claims that a hybrid solution narrows the search space and accelerates the rate of convergence during the optimization process, improving search effectiveness while still allowing the algorithm to escape from local minima.

Gomes et al. [2013] focuses on a mathematical programming approach to reactive scheduling in a make-to-order job shop with re-entrant processes and assembly. The research was based on actual data from a mold manufacturing corporation. Reactive scheduling allows a rapid response to unanticipated events in an industrial environment, such as processing delays, machine malfunctions, the delivery of new orders, or material shortage. Their future work discusses the necessity to account for the temporary unavailability of machines, changes in deadlines, or the necessity for rework due to product defects, all of which are typical in mold manufacturing.
In the scope of our study, we intend to focus simply on process scheduling using PM, although we recognize the importance of other approaches to the same issue, and especially if they comprise similar real-world scenarios.

A cloud-based cyber-physical system is presented in Mourtzis and Vlachou [2018] for adaptive shop-floor scheduling and condition-based maintenance. They demonstrate that it is possible to develop a Software-as-a-Service (SaaS) infrastructure that uses dependable real-time data for processing and analysis of shop floor data at a low cost. This infrastructure is capable of performing adaptive scheduling by generating alternatives based on a set of inputs that include demand, order-model, resource-task appropriateness, and the task sequence required. Then, using a set of criteria and weights assigned to each alternative, the utility value of each alternative is computed, and the alternative with the highest utility is selected.

Still focusing on scheduling processes in the context of this industry, Fu et al. [2019a] proposed a multi-objective pigeon inspired optimization for a Fuzzy Production Scheduling Problem considering Mould Maintenance (FPSP-MM). Using an algorithm inspired by the behavior of pigeons, the scheduling solutions compete to be the flock leader (also known as pigeon leader), and a particular crowding distance is applied to assure a decent distribution of solutions across the search space. The results show that this is a good approach to FPSP-MM compared to other approaches mentioned in the article. This research makes no mention of information gathering or, in this instance, any process mining techniques.

The same authors of the prior study, developed a three-level particle swarm optimization with variable neighborhood search algorithm for the Production Scheduling Problem with Mold Maintenance (PS-MM) [Fu et al., 2019b], referred to as TLPSO-VNS. PS-MM is a single-operation scheduling problem with numerous simultaneous machines, meaning each job has only one operation, and each workshop has several machines. It is divided into three subproblems: production scheduling, machine maintenance, and mold maintenance. Even though the TLPSO-VNS algorithm is an optimization method that is adapted to solve these subproblems, it has a high time complexity and may not be efficient for larger problems. It is mentioned that future work will focus on improving the efficiency and adaptability of the algorithm.

Jong et al. [2020] chose to integrate a combination of a *First In First Out* and *Earliest Due Date* heuristics for the scheduling in the production lines of a mold manufacturing shop floor. A GA was developed first with the objective of finding better scheduling sequences, and after that, an Ant Colony Optimization was implemented, so it could optimize the results obtained. In order to add complexity to the set of GAs, actual mold instances with four-layer components were used in this research, since the components of the lower layers cannot be constructed until the components of the layers above are complete. This approach saved 10% more time than the *Earliest Due Date* based scheduling (sorting tasks by estimated due time). Future research will incorporate multi-objective optimization and expand it to a multi-agent system, as this has been proven to be an effective method for overcoming dynamic scheduling issues by prior research.

Lee et al. [2020] approached the problem by using deep reinforcement learning, more specifically the Q-network algorithm, for the scheduling in injection mold production. The deep Q-network learned by analyzing multiple iterations of the problem of scheduling molds. After the training was complete, the performance of the proposed algorithm was evaluated using rules that minimized the overall weighted tardiness. The results show a lower total weighted tardiness than other rule-based methods.

The work of Choueiri and Santos [2021] is not related to the mold industry, but rather to multi-product scheduling via process mining. They describe a framework created with the purpose of extracting processes from event logs and exploiting the dependencies collected in the data to automate the scheduling of a particular product and its components using the resources found. Inspired by the work of Jung-Ug Kim and Yeong-Dae Kim [1996], a GA with Random Keys was used for the scheduling method. They emphasize the development of a bisection decoding approach in their study to optimize the GA from the traditional $O(n)$ decoding to $O(\log n)$.

## 3.2 Conclusion

In conclusion, the related work in the scheduling of molds' manufacturing processes demonstrates that there has been relatively few prior studies which cover the use of PM techniques to collect data for scheduling in the context of mold manufacturing. The majority of studies focus on the different approaches for the scheduling of mold's manufacturing or maintenance processes, without accounting for the use of PM as a way to gather real process flows and the duration of each activity.

The work of Choueiri and Santos [2021], provides a good foundation to the extraction of dependencies from a Petri net and the subsequent schedule, although not relating it to the mold industry nor to its constraints. The works of Wu et al. [2009] and Caballero-Villalobos et al. [2013], also offer promising approaches, but both have their limitations. None of these articles take the factory's activity into account, meaning that each schedule solely focuses on allocating the necessary jobs to the correspondent feasible machines, regardless of whether those machines are currently occupied by other manually or automatically scheduled jobs.

Given this context, we chose to build upon the work of Choueiri and Santos. This decision was influenced by a preliminary prototype that demonstrated the transferability of their techniques and ability to adapt to the operational challenges and restrictions specific to the mold industry and the Prom4Prod project. This topic is covered in more detail in future sections, specifically in Chapter 5, in which we describe the project requirements.

# Chapter 4

# Research and Development Methodologies

This chapter presents the established methodologies employed in this work. Firstly, the research methodology is presented, followed by the software development methodology, outlining the techniques used to design and implement the solution.

## 4.1 Research Methodology

Given that the outcome of this work is a software solution that involves the use of real data, including data cleaning and standardization, we employed an adaptation of the Cross Industry Standard Process for Data Mining (CRISP-DM) methodology.

CRISP-DM is a widely-used methodology for data mining and process improvement projects, designed to guide organizations through the process of extracting value from data [Chapman et al., 2000]. It has been demonstrated to be effective in a variety of industries, including manufacturing [Gellrich et al., 2019].

As illustrated in Figure 4.1[1], the CRISP-DM methodology consists of six main phases, each of which are critical to the project's success. In this work, we followed the methodology as outlined below. The detailed explanation of the development of our approach can be found further in the document (Chapter 5).

1. **Business understanding:** Consulted with a key partner from the Prom4Prod project, an expert in the mold industry, to comprehend industry constraints, business logic, challenges in mold manufacturing, customer demands in the field, as well as identifying the problem and its relevant context (e.g., what is the problem, why it is important, and who it affects);
2. **Data understanding:** Addressed data recorded on production floors, in order to understand its structure, content, and any problems that could exist;

[1]Designed using images from Flaticon.com.

3. **Data preparation:** Consisted on preprocessing the data gathered, in order to tackle challenges like data cleaning, filtering, and standardization to gain insights into the necessary data transformations for the project. Additionally, the Inductive Miner Infrequent PM algorithm was used to extract process knowledge from specific parts used in the mold families, resulting in a detailed understanding of the individual processes involved in the production of different mold parts;

4. **Modeling:** Included the development of a recursive algorithm for sequence extraction, another algorithm to clean this sequence, the manipulation of the data gathered from the *event logs*, and the implementation of a GA for generating schedules;

5. **Evaluation:** Consisted in the assessment of the sequence extraction method and the GA to verify their functionality and feasibility in accordance with the data;

6. **Deployment:** Integration of this work results into the Prom4Prod platform, along with the execution of any required modifications to align with the application's business logic.

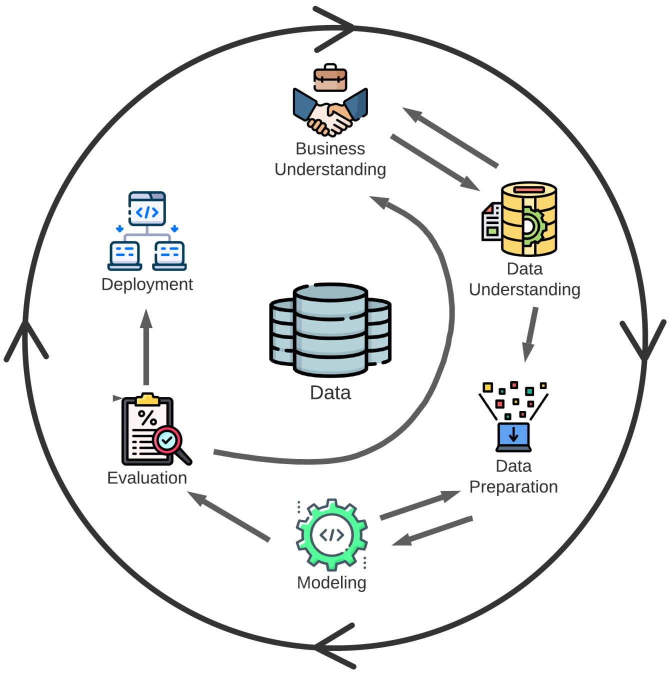


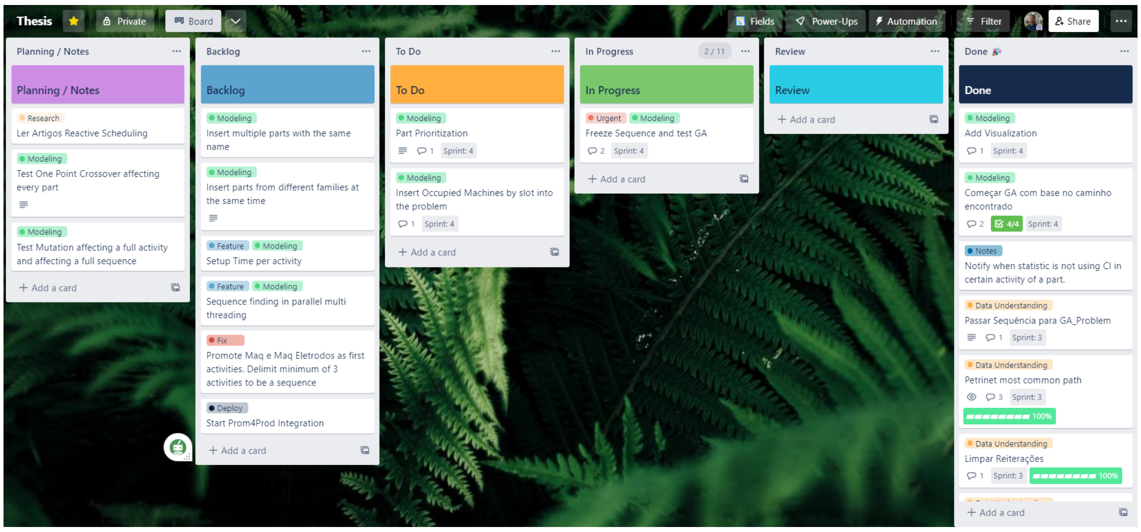
Figure 4.1: Phases of CRISP-DM Model adapted from [Chapman et al., 2000].

The stages of the CRISP-DM methodology were not strictly linear in this work, as some steps underwent revision as we explored different approaches to extract the desired data, or even adapted the process based on the evolving understanding of the data and the industry context. The iterative nature of the project allowed us to learn more about the data and the specific nuances of the industry, leading to valuable insights that influenced the way certain steps were conducted.

## 4.2 Software Development Methodology

As a software development methodology, we have chosen to use Kanban which is an Agile method for software development that enhances project management by providing a visual representation of the work via a "Kanban Board" [Alaidaros et al., 2021]. Typically, a Kanban board consists of columns such as To-do, Doing, and Done, although it can be tailored to align with the specific workflow states of the team. This choice allows us to effectively track and manage the progress of our tasks.

The main rationale behind the choice of this methodology was to achieve flexibility in our development process. By adopting an iterative development, we intended to continually refine and improve our solution. To ensure an effective communication, we scheduled weekly meetings where the Kanban methodology played a crucial role in displaying the current project status, showcasing small demos, and discussing potential changes.



Planning / Notes
- Research — Ler Artigos Reactive Scheduling
- Modeling — Test One Point Crossover affecting every part
- Modeling — Test Mutation affecting a full activity and affecting a full sequence

Backlog
- Modeling — Insert multiple parts with the same name
- Modeling — Insert parts from different families at the same time
- Feature, Modeling — Setup Time per activity
- Feature, Modeling — Sequence finding in parallel multi threading
- Fix — Promote Maq e Maq Eletrodos as first activities. Delimit minimum of 3 activities to be a sequence
- Deploy — Start Prom4Prod Integration

To Do
- Modeling — Part Prioritization — 1 — Sprint: 4
- Modeling — Insert Occupied Machines by slot into the problem — 1 — Sprint: 4

In Progress 2 / 11
- Urgent, Modeling — Freeze Sequence and test GA — 2 — Sprint: 4

Review

Done
- Modeling — Add Visualization — 1 — Sprint: 4
- Modeling — Começar GA com base no caminho encontrado — 2 — 4/4 — Sprint: 4
- Notes — Notify when statistic is not using CI in certain activity of a part.
- Data Understanding — Passar Sequência para GA_Problem — 1 — Sprint: 3
- Data Understanding — Petrinet most common path — 3 — Sprint: 3 — 100%
- Data Understanding — Limpar Reiterações — 1 — Sprint: 3 — 100%

Figure 4.2: Trello Kanban board in 4th week of development.

To implement our Kanban methodology, we used Trello[2] as our preferred Kanban board platform, as shown in Figure 4.2. Our Trello board consisted of six columns, namely: Planning/Notes, Backlog, To Do, In Progress, Review, and Done. This setup allowed us to effectively manage and track our tasks throughout their life-cycle, from initial planning and prioritization to completion and review. The clear categorization provided by these columns enabled us to visualize the progress of each task and streamline our workflow accordingly.

[2]`https://trello.com`

# Chapter 5

# Development

In this chapter, we begin by describing the main functional and non-functional software requirements and the Prom4Prod platform's general software architecture, as well as our solution's particular architecture, known as the Process Scheduling module. We define each individual stages that comprise this module and provide an insight into the tasks performed at each stage.

For a better understanding of the solutions' workflow, we intend to use a series of examples to better convey this explanation.

## 5.1 Requirements

Functional and non-functional requirements are essential to align our work with the current objectives (Section 1.3). These specifications form the basis of our work, which aims to produce a solution that assists mold manufacturing companies increasing the efficiency and efficacy of their scheduling processes.

The primary objective of this project is to develop a feature for the Prom4Prod project that automates the schedule of mold manufacturing processes in the mold industry using PM to gather a more realistic business process (compared to conventional approaches in Business Process Management (BPM)). To guide the development of this feature, a set of objectives were established and converted into functional requirements.

Functional requirements are an important part of software specification, as they help to describe the functionalities that are expected. In addition to functional requirements, non-functional requirements should also be taken into account, as they specify the constraints and qualities of the desired solution.

In the following sections, we describe the decisions made and the defined requirements.

### 5.1.1 Functional Requirements

At the start of the design process for the Prom4Prod project, a set of objectives were established, namely:

1. The system should allow users to input mold manufacturing data, such as the characteristics of the molds parts to be produced, and the available time frame;
2. The system must be able to access event log data retrieved from the factory floors in order to discover business processes executed;
3. The system should generate optimal schedules for the activities belonging to the production processes of the molds, taking into account the available resources (e.g., machines used) found in previous business processes and time frame given by the user;
4. The system should provide visualizations of the calculated schedules (i.e., Gantt charts or other graphical representations);
5. The system should allow users to modify the input data and re-calculate schedules as needed;
6. The system should take into account prior schedules made in the factory floor, meaning that machines can be already occupied in the given time frame;
7. Each process activity must be appointed to an available feasible machine, meaning that each part can only be scheduled to a machine where it can be executed;
8. All process activities must comply to the proper production sequence.
9. Each machine can only carry out a single process activity at a time.
10. If a process activity is delayed, the dependent activities in the production sequence must wait.

### 5.1.2 Non-functional Requirements

As the Prom4Prod project involves a large number of participants, non-functional requirements were not formally defined in its initial stages. However, it is possible to infer a number of non-functional requirements based on the needs of the solution:

1. To ensure data integrity and security, access to the application should be restricted. Thus, only users who are authenticated and authorized in the system database should have access to the platform and the information contained within it. Any unauthorized users should be denied access to the platform and its data to maintain data confidentiality and prevent potential security breaches.

2. The optimization solution must be maintainable, with access to clear documentation and a well-designed architecture that allows for easy updates and future expansions. This report, along with the technical report of the Prom4Prod project, serves as the documentation for the system's design and development.

3. The solution must be developed using Python, since it is adopted by the Prom4Prod project;

Having covered the functional and non-functional requirements, we can now turn our attention to the description of both the Prom4Prod architecture and the solution architecture.

## 5.2 Architecture

The Prom4Prod project has been developed as a decision support system specifically tailored for the plastics injection mold industry. Its primary objective is to provide analytical insights, identify bottlenecks and deviations in the processes, and support decision-making regarding resource allocation within engineer-to-order processes in the factory floors. With a focus on using PM, the Prom4Prod project seeks to help stakeholders make more informed decisions.

Our contribution comes in the form of the Process Scheduling module. This module integrates our work into the existing project framework. In the following sections, we look at the architectural aspects of both the Prom4Prod project and the Process Scheduling module, as well as the design decisions and components that incorporate the solution.

### 5.2.1 Prom4Prod architecture

The Prom4Prod platform has been designed with a three layer architecture, as depicted in Figure 5.1. The presentation layer acts as the user interface, allowing users to interact with the system. The business logic layer, located beneath the presentation layer, incorporates an Application Programming Interface (API) to facilitate communication with external applications, which processes the input provided by the presentation layer, and consumes the collected data through IoT sensors, API or other third-party applications. This layer applies various algorithms and techniques, with Process Mining (PM) as its main resource for analysis and decision-making. Additionally, it communicates with the Local Data Collector, OnFlow, to transmit IoT data for Key Performance Indicators (KPIs) and raw data management. Lastly, the data access layer is responsible for storing and managing PM results, ensuring its integrity and availability.

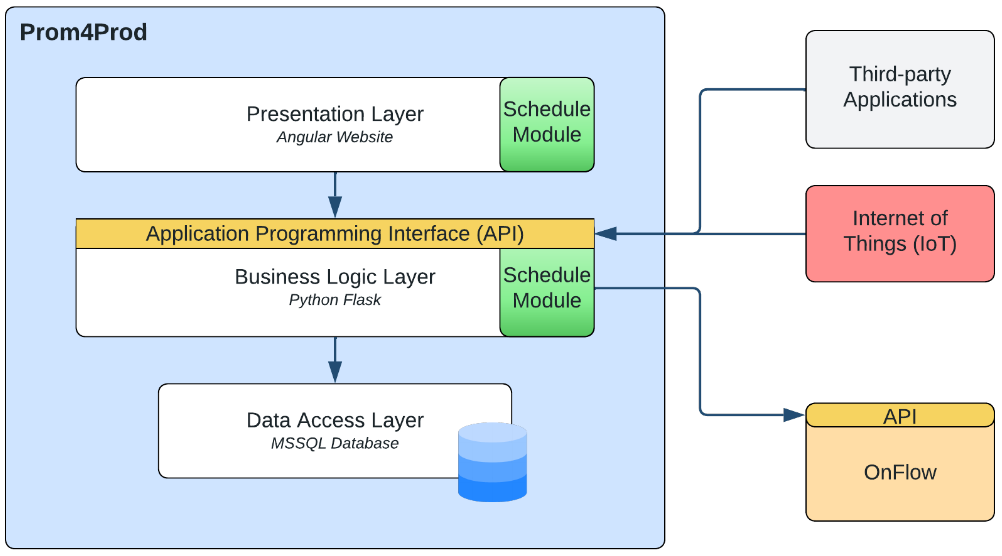


Figure 5.1: Prom4Prod 3-layer architecture.

### 5.2.2 Process Scheduling module architecture

The Process Schedule module depicted in Figure 5.1 is incorporated into the Prom4Prod platform via an interface page within the presentation layer that allows the user to input the required information to schedule the desired mold pieces. The backend of the module is incorporated into the Business Logic Layer and shares an API entrypoint access with the backend of the platform, allowing for the communication between the system layers.

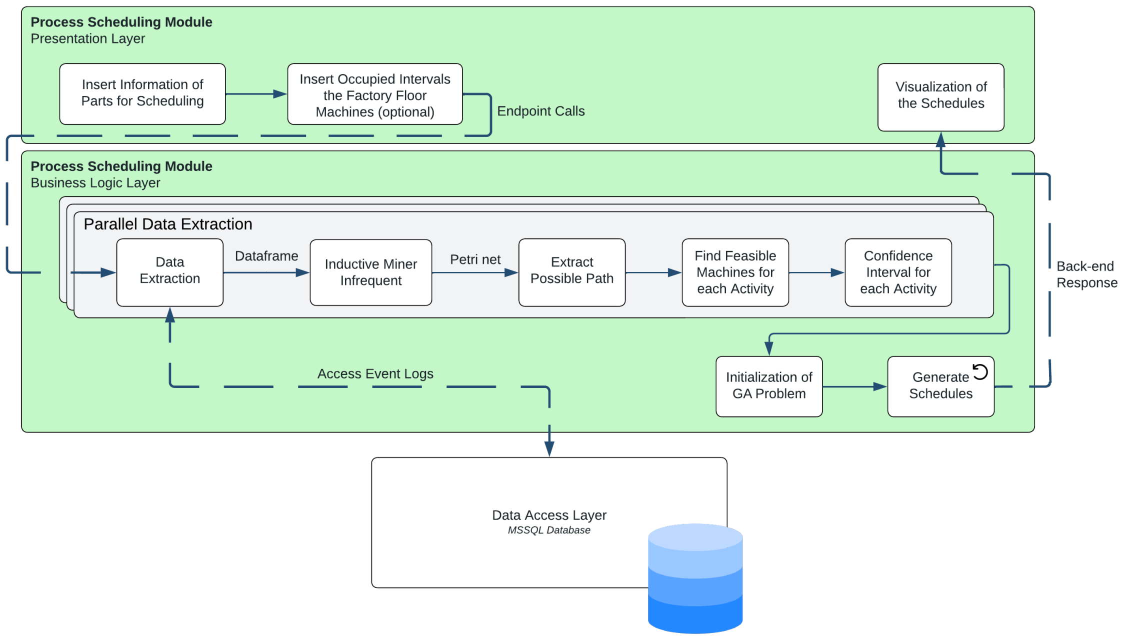

Process Scheduling Module
Presentation Layer
Insert Information of Parts for Scheduling
Insert Occupied Intervals the Factory Floor Machines (optional)
Endpoint Calls
Visualization of the Schedules
Process Scheduling Module
Business Logic Layer
Parallel Data Extraction
Data Extraction
Dataframe
Inductive Miner Infrequent
Petri net
Extract Possible Path
Find Feasible Machines for each Activity
Confidence Interval for each Activity
Back-end Response
Access Event Logs
Initialization of GA Problem
Generate Schedules
Data Access Layer
MSSQL Database

Figure 5.2: Process Scheduling module architecture.

Figure 5.2 provides an overview of the module's architecture. The Presentation Layer serves as the bridge between user interaction and the Business Logic

Layer, relaying information from the input forms. Inside the Business Logic Layer, a process named "Parallel Data Extraction" retrieves metrics from both the database and the Inductive Miner Infrequent algorithm. Once the necessary data for scheduling the required parts is extracted, it is transformed into a Genetic Algorithm (GA) problem format. Following this, the GA is executed. The schedules are then communicated to the frontend for display. Further sections will cover the specifics of each step of this process, how the GA is executed and what schedules are outputted.

## 5.3 Parallel Data Extraction

The parallel data part extraction process plays a crucial role in gathering the necessary information and requirements for scheduling parts of a mold.

As previously explained in Section 2.1, the mold industry is characterized by unpredictable processes due to the unique nature of each manufacturing process. A single mold part can undergo multiple stages, such as CNC machining, polishing, or erosion, before progressing further in the manufacturing process. This iterative loop continues until the part meets the required specifications.

To extract meaningful data from the event logs, we employed a filtering approach based on "family molds". This involved identifying molds within the same production area, possessing the same number of cavities, exhibiting similar weight characteristics or number of parts, and, if necessary, matching specific customer requirements.

For instance, when scheduling a part for a mold that produces car pedals, we extract the necessary information from the car pedal mold family to obtain precise details, including durations, machines used, and activities performed.

The parallel data extraction process involves a sequential execution of multiple steps. These sequential processes for each part can be carried out in parallel. For instance, if there are multiple parts to be scheduled, the data extraction process for each part can be performed concurrently, allowing for efficient and parallelized data extraction, as visually indicated in gray in Figure 5.2.

### 5.3.1 Data Extraction

In order to extract the necessary data, we gather the required inputs from the frontend (i.e., presentation layer components). This includes part requirements (such as part code, due date, and schedule priority), mold information (including mold identification and its family), and the time intervals during which the machines on the factory floor are occupied.

Once this data is collected (either by retrieving it from a CSV file or a database), we initiate a data cleaning process consisting in data standardization and event log simplification. Event logs are generated based on the activation of machinery sensors when a process is executed. Upon scrutinizing the data, a recurring pattern emerged: multiple event logs often originated for the same process, each log spanning seconds or minutes and featuring overlapping time intervals. To tackle this issue, a two-fold approach was adopted.

First, event logs were sorted by beginning and ending times. Subsequently, logs with corresponding criteria, such as the same mold, machine, and activity, were subsequently merged. By aggregating the logs, the concatenated versions included the entire duration of the mining process, with the earliest start date and the latest end date encapsulated within each log.

Table 5.1 and Table 5.2 present an example of a final version of the event log for parts 100 and 200 of molds, respectively. These tables showcase specific instances of event logs extracted from the factory floor, filtered based on the corresponding mold family (e.g., Car Pedals) and part identification. The columns in these tables provide the following information: *Part ID* indicates the identification number of the part, *Mold ID* represents the identification number of the mold associated with the part, *Activity* denotes the specific activity or operation performed on the part, *Machine ID* corresponds to the identification number of the machine involved in the activity, *Start* corresponds to the start date and time of the activity, and *End* denotes the end date and time of the activity.

Table 5.1: Event log fragments for part 100.

| Part ID | Mold ID | Activity | Machine ID | Start | End |
|---|---|---|---|---|---|
| 100 | 350 | CNC | m1 | 15/05/2023 08:10 | 15/05/2023 08:24 |
| 100 | 350 | Erosion | m4 | 15/05/2023 08:25 | 15/05/2023 08:30 |
| 100 | 350 | Polishing | m6 | 15/05/2023 08:32 | 15/05/2023 08:40 |
| 100 | 350 | Erosion | m5 | 15/05/2023 08:43 | 15/05/2023 08:58 |
| 100 | 350 | Polishing | m6 | 15/05/2023 09:05 | 15/05/2023 09:17 |
| 100 | 477 | Electrodes | m3 | 15/05/2023 08:05 | 15/05/2023 08:10 |
| 100 | 477 | Erosion | m6 | 15/05/2023 08:12 | 15/05/2023 08:22 |
| 100 | 477 | Polishing | m6 | 15/05/2023 08:22 | 15/05/2023 08:30 |
| 100 | 399 | CNC | m2 | 15/05/2023 08:10 | 15/05/2023 08:26 |
| 100 | 399 | Erosion | m5 | 15/05/2023 08:29 | 15/05/2023 08:39 |
| 100 | 399 | Polishing | m7 | 15/05/2023 08:40 | 15/05/2023 08:50 |

Table 5.2: Event log fragments for part 200.

| Part ID | Mold ID | Activity | Machine ID | Start | End |
|---|---|---|---|---|---|
| 200 | 1012 | CNC | m2 | 15/05/2023 14:00 | 15/05/2023 14:18 |
| 200 | 1012 | Erosion | m4 | 15/05/2023 14:20 | 15/05/2023 14:25 |
| 200 | 1012 | CNC | m1 | 15/05/2023 14:26 | 15/05/2023 14:48 |
| 200 | 1012 | Erosion | m5 | 15/05/2023 14:50 | 15/05/2023 14:55 |
| 200 | 1012 | Polishing | m6 | 15/05/2023 14:56 | 15/05/2023 15:10 |
| 200 | 880 | Electrodes | m3 | 15/05/2023 14:00 | 15/05/2023 14:15 |
| 200 | 880 | Erosion | m6 | 15/05/2023 14:16 | 15/05/2023 14:21 |
| 200 | 880 | Polishing | m7 | 15/05/2023 14:24 | 15/05/2023 14:40 |

This data frame is passed on to the subsequent steps of the sequence, so the desired metrics can be used. These metrics include the resources that were previously used, the durations of the activities, and the sequence of activities performed.

### 5.3.2 Inductive Infrequent Miner

The Inductive Miner is a process discovery algorithm used in PM employed to extract infrequent patterns from process event logs [Leemans et al., 2015].

A crucial feature of the Inductive Infrequent Miner is the ability to interact with the noise threshold. By adjusting the noise threshold, users can control the amount of noise generated within the Petri net, thus gaining greater control over the mining process [Leemans et al., 2013]. Consequently, we have chosen to use this particular variant of the Inductive Miner, as it empowers end-users to simplify or increase the complexity of the resulting Petri net.

Our purpose is to use this Petri net to be able to understand what are the possible paths of the process to be scheduled. We can also visualize the frequency of activities in the Petri net itself and use this information to later find a relevant path, as discussed in the following section.

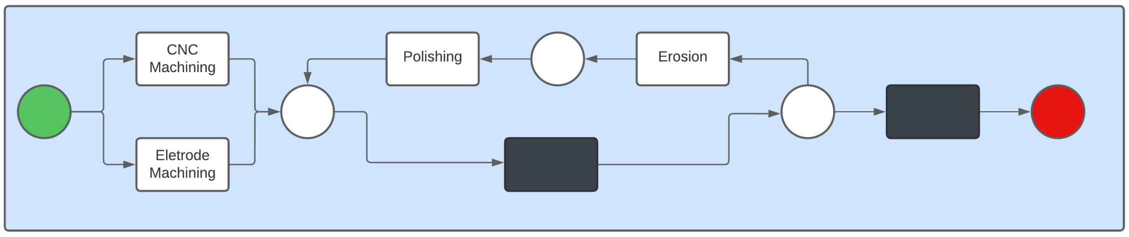


Figure 5.3: Petri net model of part 100 manufacturing processes.

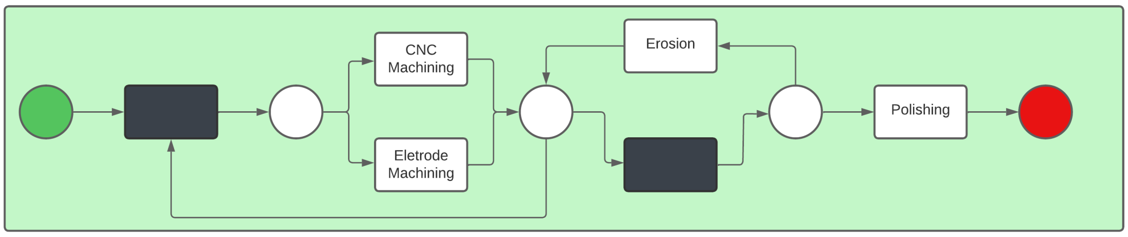


Figure 5.4: Petri net model of part 200 manufacturing processes.

Taking reference from the event log fragments provided in Table 5.1 and Table 5.2, corresponding Petri nets representing the processes of part 100 and part 200 can be observed in Figures 5.3 and 5.4, respectively. The analysis of the Petri nets reveals that certain activities, such as "Erosion" in Part 200, are optional and/or can be replicated, while others are essential for the completion of the process, like "Polishing" in Part 200.

### 5.3.3 Sequence Extraction

To conduct a sequence extraction from the Petri net, we decorate it with frequency information, making it possible to visualize and retrieve data from the frequency of transitions within the system. In Figure 5.5 we can observe an example of the Part 200 processes Petri net decorated with frequency information.

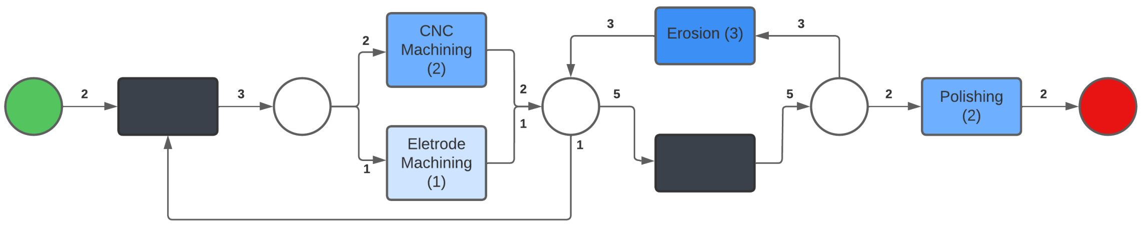


Figure 5.5: Frequency of part 200 processes.

Our objective is to extract a meaningful and feasible path from the Petri net while avoiding infinite loops. To achieve this, we explore the Petri net by searching for each possible option after every transition, using a recursive approach.

For instance, let us consider the probabilities of activities in the Petri net of Figure 5.5. "CNC Machining" may have a probability of $\frac{2}{3}$ of being selected, while "Electrode Machining" has a probability of $\frac{1}{3}$. Once an activity is chosen to be part of the sequence, we initiate the search again, starting from that transition. However, to discourage excessive repetition, we penalize the probability of the selected activity by reducing it by half.

By applying this penalty, we decrease the likelihood of repeatedly selecting the same activity in subsequent iterations. This promotes a more diverse and varied sequence by discouraging excessive repetition of activities.

This method is particularly useful when dealing with a Petri net derived from a large event log, such as a log containing information about the completion of 100 registered parts. In such cases, there may be a significant amount of activity repetition. If we were to select only the most common activities, we would overlook the less frequently performed ones, which are still crucial for the overall process. Taking the "Erosion" activity as an example, even if it was less frequently observed in the event log, it would still play a vital role in the process. Therefore, by incorporating randomness into the selection process based on assigned probabilities, we ensure that even the less common but essential activities have a chance of being included in the resulting sequence.

Examples of the possible sequences extracted from the Petri nets for parts 100 and 200 can be observed in Figures 5.6 and 5.7. These sequences can then be used as a foundation for scheduling these parts.

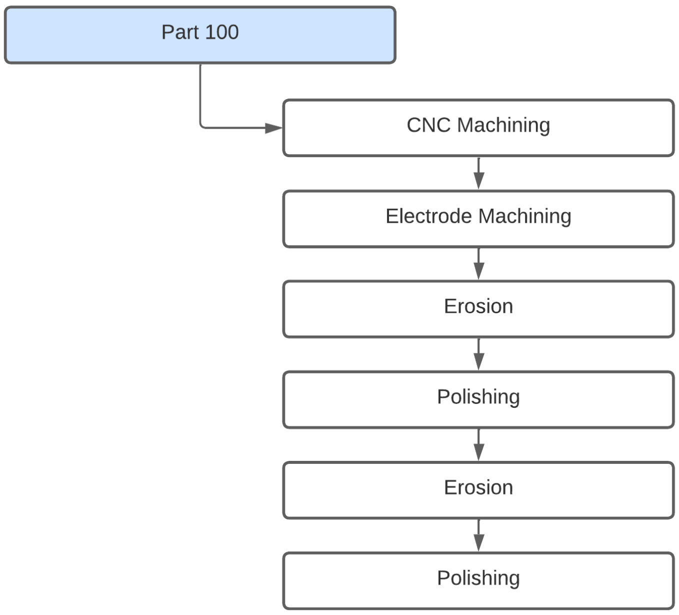


Figure 5.6: Sequence extracted for part 100.

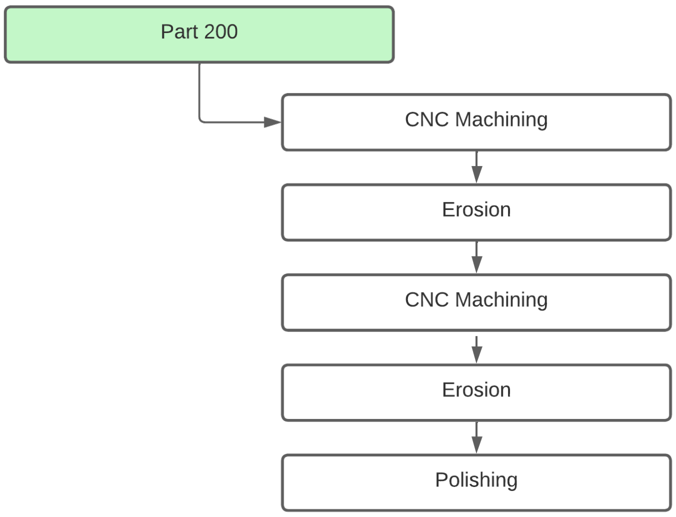


Figure 5.7: Sequence extracted for part 200.

Furthermore, this sequence extraction approach can meet scenarios with parallel activities. The method maintains a neutral stance, allowing for the possibility of activities that can be executed in parallel. This flexibility aligns with real-world scenarios observed by the PM algorithms, where certain activities can occur simultaneously. The approach considers the probabilities assigned to each transition, regardless of their potential parallel execution, resulting in a comprehensive and adaptable sequence extraction process.

### 5.3.4 Assignment of Feasible Machines and Durations

With the retrieved sequence, we can use the corresponding data frame to extract relevant metrics that will be used in scheduling. For instance, we can determine the feasible machines available for each activity. This information allows us to understand which machines are suitable for executing a particular task.

In addition, we extract the processing times (i.e., durations) associated with each activity from the data frame. These durations reflect the time required to complete each task. By employing confidence intervals, we can estimate values for pessimistic, realistic, and optimistic scenarios based on the extracted durations. Confidence intervals allow us to quantify the variability and uncertainty associated with each activity's duration, providing a more comprehensive understanding of the time required for process completion.

The usage of confidence intervals to estimate different scenario values are further clarified in the next section, where we explore how these metrics contribute to effective scheduling strategies.

Table 5.3: Metrics for the activities in the sequence for part 100.

| Activity | Feasible Machines | Durations (minutes) |
|---|---|---|
| CNC Machining | m1, m2 | 14, 16 |
| Electrode Machining | m3 | 5 |
| Erosion | m4, m5, m6 | 5, 10, 10, 15 |
| Polishing | m6, m7 | 8, 8, 10, 12 |

Table 5.4: Metrics for the activities in the sequence for part 200.

| Activity | Feasible Machines | Duration (min) |
|---|---|---|
| CNC Machining | m1, m2 | 18, 22 |
| Erosion | m4, m5, m6 | 5, 5, 5 |
| Polishing | m6, m7 | 14, 16 |

Table 5.3 shows metrics for activities in the sequence for part 100. It lists feasible machines and their corresponding durations in minutes. For example, "CNC Machining" works on machines *m1* and *m2* for 14 and 16 minutes respectively. "Electrode Machining" uses *m3* for 5 minutes. "Erosion" involves using *m4*, *m5*, and *m6* with durations of 5, 10, and 15 minutes. "Polishing" happens on *m6* and *m7* for 8 and 12 minutes.

Table 5.4 displays metrics for activities in the sequence for part 200. Similar to the previous table, it lists feasible machines and durations. "CNC Machining" uses *m1* and *m2* for 18 and 22 minutes. "Erosion" involves *m4*, *m5*, and *m6* for 5 minutes each. "Polishing" is done on *m6* and *m7* for 14 and 16 minutes.

These tables provide an overview of the extracted metrics from the data frame for each sequence. It is important to note that there are fewer unique machines compared to durations, given that machines can be used multiple times across the event logs. Additionally, each duration corresponds to the time extracted from individual captured logs. For instance, in the case of the "Erosion" activity in Table 5.3, machine *m5* is used twice, resulting in two separate duration entries for that specific machine and activity.

### 5.3.5 Confidence Intervals

By retrieving the durations for each activity of each part from the data frame, we can then follow the approach of Choueiri and Santos [2021]. This approach involves generating three processing times based on the collected durations for each activity, an optimistic, a realistic and a pessimistic scenario. This allows industrial managers to be better equipped and make more informed decisions. For instance, when discussing due dates with clients, managers can rely on the pessimistic scenario, providing a safe delivery estimate. On the other hand, when dealing with suppliers, the optimistic scenario can be used. This has the potential to assist in planning material orders and potentially reducing the risk of production delays that might arise from material shortages.

The scheduling process relies on confidence intervals derived from the retrieved durations. The optimistic scenario is built with the shortest feasible processing times by using the lower values of the confidence intervals. For the realistic scenario, the mean of the processing times are employed, reflecting the most probable durations. In the pessimistic scenario, all devices operate at the maximum feasible time, therefore the upper value of the intervals is used. Depending on the characteristics of the collected data, the method used to obtain these values could be either a mean confidence interval or a bootstrap interval [Efron, 1979; Lilja, 2000].

Figure 5.8 illustrates a decision chart adapted from the original paper [Choueiri and Santos, 2021], illustrating the process of selecting an appropriate method for generating a confidence interval. The flowchart evaluates the necessary assumptions for conducting a confidence interval. The first decision point, indicated by "$n > 70$", takes into account the Central Limit Theorem, which states that the sum of $n$ independent and identically distributed random variables approximates a normal distribution. The chosen threshold of 70 cases aligns with the approach of Choueiri and Santos [2021], as it strikes a balance between a sufficiently large sample size and practical considerations.

If this assumption is not satisfied, the normality of the data is evaluated using the Shapiro-Wilk test [Shapiro and Wilk, 1965]. The null hypothesis ($H_0$) of the test is that the sample is drawn from a normal distribution. By examining the resulting *p-value*, if the *p-value* is greater than 0.05 ($\alpha > 0.05$), it suggests that the data can be reasonably assumed to follow a normal distribution, and a confidence interval can be generated using traditional methods. However, if the *p-value* is less than or equal to 0.05, indicating a significant departure from normality, the bootstrap interval will be employed as an alternative approach.

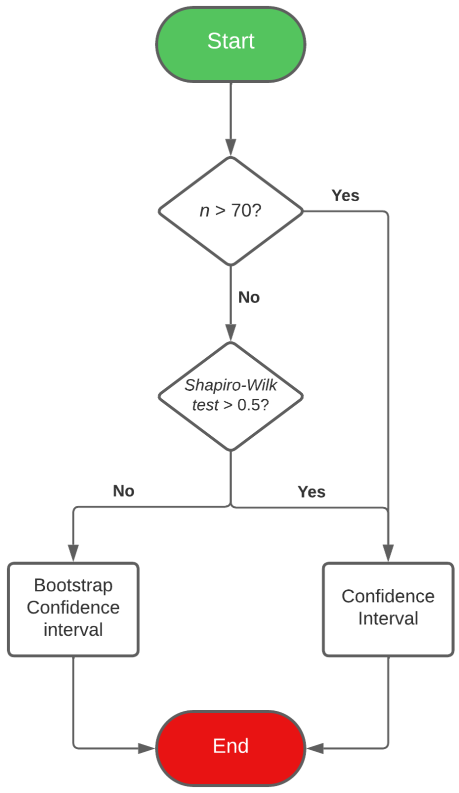


Figure 5.8: Decision flow for the usage of confidence intervals.

The mean confidence intervals states that $(1-\alpha)\%$ of the durations would fall on the interval. The $\alpha$ is set as 5% as usual. Since we do not know the population mean $\mu$ nor the variance $\sigma^2$, the interval is constructed according to the *t-student* distribution [Lilja, 2000]:

$$(\bar{x} - t_{1-\alpha/2;n-1}\frac{S}{\sqrt{n}}, \bar{x} + t_{1-\alpha/2;n-1}\frac{S}{\sqrt{n}})$$

where $\bar{x}$ is the sample mean, $t_{1-\alpha/2;n-1}$ is the critical value from the t distribution with $n-1$ degrees of freedom, such that it leaves an area of $t_{1-\alpha/2}$ in the tails on each side of the distribution, $\alpha$ is the significance level which determines the confidence level of the interval (e.g., for a 95% confidence interval, $\alpha = 0.05$). $S$ is the sample standard deviation and $n$ is the sample size.

If the assumptions necessary to construct the parametric confidence interval are not met, we use the bootstrap confidence interval [Efron, 1979]. The bootstrap interval relies on resampling techniques to estimate the sampling distribution and construct the confidence interval, making it suitable for non-normal data. The first stage is to generate multiple samples of the same size (with replacement) from the collected durations. Each new resample has a mean value, denoted by $\bar{X}^*$. To construct the confidence interval, we must determine how much the distribution of $\bar{X}$ deviates from $\mu$:

$$\delta = \bar{X} - \mu$$

The bootstrap principle states that we can approximate $\delta$ by:

$$\delta^* = \bar{X}^* - \bar{X}$$

Therefore, it is possible to compute all resampled valued of $\delta^*$ , and sort them in non-decreasing order. The bootstrap confidence interval is then:

$$[\bar{X} - \delta_a^*, \bar{X} + \delta_b^*]$$

where $a$ and $b$ are percentiles of the collected $\delta^*$ values. For example, if we want a 90% confidence interval, we look at the 5th percentile ($a$) and the 95th percentile ($b$). The value at the 5th percentile, $\delta_a^*$, represents the lower boundary of our confidence interval, and the value at the 95th percentile, $\delta_b^*$ represents the upper boundary.

Furthermore, in order to gather the processing times for all three scenarios, we must determine whether the data contains more than three observations. In instances when the data size is less than three, we use the average of the available data as a substitute for all three processing times. However, it is important to note that a warning will be issued to the user, indicating that the processing times were generated based on the average value.

Additionally, if the number of cases is less than 10, the sample size may be insufficient to generate precise confidence intervals. Therefore, a warning message is provided to the user, emphasizing that the available data may not yield highly accurate confidence intervals due to the limited sample size.

Table 5.5: Processing times gathered for the scenarios.

| Scenarios | Bootstrap CI | CI |
|---|---|---|
| optimistic | $\bar{x} - \delta_a^*$ | $\bar{x} - t_{1-\alpha/2;n-1}\frac{S}{\sqrt{n}}$ |
| realistic | $\bar{x}$ | $\bar{x}$ |
| pessimistic | $\bar{x} + \delta_b^*$ | $\bar{x} + t_{1-\alpha/2;n-1}\frac{S}{\sqrt{n}}$ |

Table 5.5 provides a summary of the processing times gathered for the scenarios using both confidence interval formulas. The table distinguishes between the optimistic, realistic, and pessimistic scenarios. In the optimistic scenario, the lower value of the confidence interval represents the lowest duration that an activity can take with 95% certainty. The realistic scenario is defined by the mean duration found in the activity, representing the most "probable" processing time. On the other hand, the pessimistic scenario is characterized by the upper value of the confidence interval, indicating the highest duration the activity can have with 95% certainty.

Once the processing times for each scenario are calculated, the resulting data is then used in a Genetic Algorithm (GA) problem. The GA problem and its exploration is discussed in more detail in the next section.

## 5.4 Genetic Algorithm

The purpose of this section is to provide thorough understanding about the GA used in this study, including its functionality, operation, and output. We begin by describing the problem that serves as input to the GA. Following that, we look into the structure of the individuals, showcasing the genotype and phenotype conversion processes, the variation operators used, the selection of parents, fitness evaluation, and other pertinent aspects.

By exploring these elements, we aim to provide a detailed insight into the functionality, operation, and output of the GA employed in this research.

### 5.4.1 Setup of the Genetic Algorithm Problem

The objective of the GA is to obtain an optimal and feasible solution to a given problem. Consequently, it is crucial to construct the problem by providing the necessary context and resources. This enables the creation of a population that can effectively search for and converge towards the desired solution.

Therefore, the problem is constructed by specifying several key elements. These include the total number of machines present on the factory floor, the intervals during which these machines are occupied, the parts that need to be scheduled, the desired sequence in which these parts should be manufactured, and the activities within this sequence. Additionally, for each activity, a list of feasible machines is provided, along with the corresponding processing times for each scenario. By defining these parameters, the problem is effectively set up to facilitate the implementation of the GA.

Furthermore, it is important to note that the GA is executed three times, once for each scenario. This approach allows for the exploration and optimization of solutions specific to each scenario, thereby providing a comprehensive analysis of the problem.

### 5.4.2 Population Initialization

The population of the GA is randomly initialized by creating a specified number of individuals. Each individual is composed of a chromosome and an initial fitness value, representing a potential solution.

The initialization process involves generating a chromosome for each individual in the population. For each order (i.e., the parts ordered for scheduling), a sequence of activities is created. The machine for each activity is randomly chosen from the feasible options, ensuring randomness and exploration. Additionally, the priority for each activity is assigned a random value within a specified range, taking into account the part's priority.

Random initialization helps ensure diversity within the population. By assigning random values to genes, different combinations are explored, searching a wide range of potential solutions and discovering different regions. It also prevents bias by starting with a randomly generated initial population, increasing the likelihood of finding better solutions [Eiben and Smith, 2015].

### 5.4.3 Individuals Representation

The representation of individuals in our GA takes the form of a structured entity known as a genotype. This genotype operates as a structure for each individual, enabling their evaluation within the algorithm. Similar to what is observed in nature, the genotype of an individual corresponds to its phenotype. In the context of our project, the phenotype is manifested as a potential schedule.

Within each individual's genotype, there exists a set of chromosomes, where the count of chromosomes matches the number of parts to be scheduled. Each of these chromosomes contains a specific number of genes, mirroring the total count of activities each part needs to undergo.

The genotype is implemented using a random keys approach [Gonçalves and Resende, 2018]. Each activity (i.e., each gene) is defined by the machine ID where the activity will be processed and a randomly assigned value ranging from 0 to 1, which determines its prioritization within the machine. Taking the data from the examples of previous steps (Figures 5.6 and 5.7 and Tables 5.3 and 5.4), an example of the individual genotype is presented in Figure 5.9.

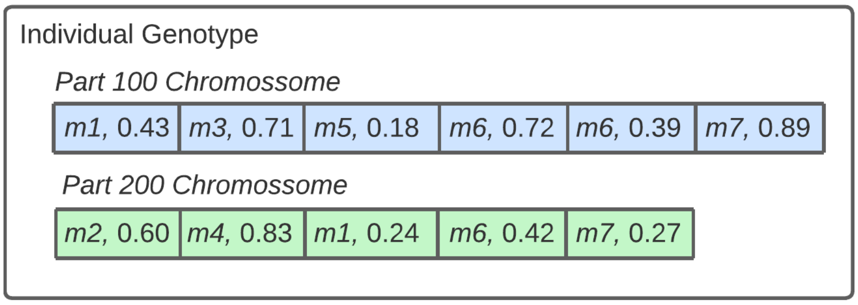


Figure 5.9: Example of an individual's genotype.

#### Genotype to Phenotype Mapping

The genotype to phenotype mapping is a crucial step in the GA, as it involves converting the genetic information in the individual's genotype (chromosomes) into a meaningful representation known as the phenotype (our schedule). In our approach, this transformation is achieved through the allocation of activities from the genotype to the corresponding machine queues based on their machine IDs and priorities.

To illustrate this process, let us consider the example of an individual's genotype shown in Figure 5.9.

To convert this genotype into a phenotype, we start by mapping the activities to their respective machine queues. Each machine has its own queue, and activities are placed in the queues based on their machine ID and priority values. Higher priority activities are positioned at the front of the queue. An example of this mapping can be seen in Figure 5.10.

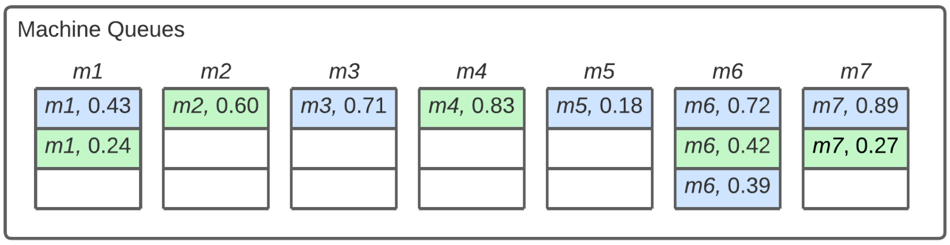


Figure 5.10: Machine queues from the genotype conversion.

For instance, let us consider the activity for part 100 in the genotype: The first activity is assigned to machine *m1* with a priority of *0.43*. We locate machine m1's queue and insert this activity with its corresponding priority into the queue. The same process is repeated for all activities in the genotype until all parts are placed in their respective machine queues.

Once the mapping is completed, each queue is iterated, and the activities are scheduled according to their priority order, while considering if their parent activities have already been scheduled. If a parent activity has not yet been scheduled, the activity is skipped during the iteration. This process allows for parallel executions, where activities can have multiple children and parents. Consequently, it creates a scenario where parts compete for their scheduling while maintaining the feasibility of their respective sequences.

The result of this conversion is a schedule where the two parts to be scheduled are accommodated simultaneously whenever possible, sometimes competing for the same resources. An example can be seen in Figure 5.11, where this schedule is recreated for the realistic scenario where the time for each activity is computed using the average processing time, as mentioned in Section 5.3.5.

In our scheduling process, we employ timeslots as a mechanism to discretize time, converting dates and durations into fixed time intervals. Each timeslot corresponds to a distinct timeframe, enabling us to break down time into manageable units. This simplifies the process of translating genetic information between the genotype and phenotype representations and streamline the insertion of occupied machines into the GA. In the context of our example, we have set the time slot granularity to 1 minute, in order to provide a detailed view of the scheduling process and the relative timings of activities. However, as a default approach, we employ a 5-minute granularity for our timeslots. This choice was guided by our analysis of real activity durations within the database, which informed us of the most suitable granularity for our purposes.

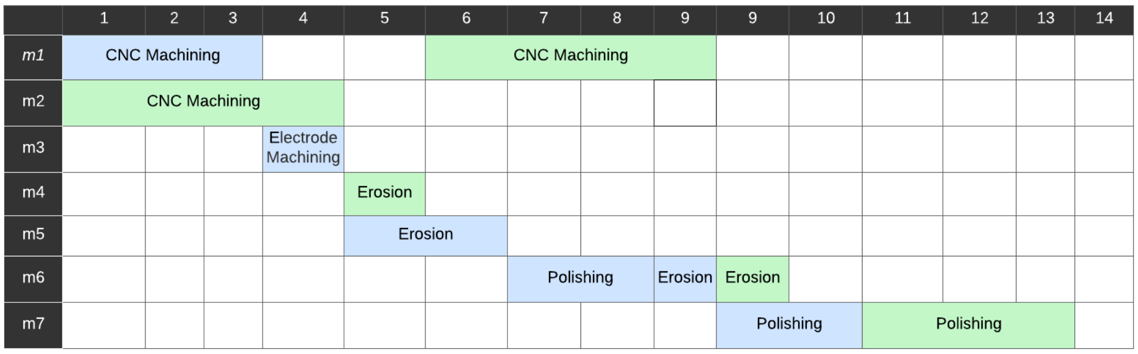


Figure 5.11: Example of the generated schedule.

In this schedule, we observe that parts 100 and 200 are scheduled concurrently at certain time slots. As a result, they may compete for the same machines or resources during those time intervals. The competition allows the GA to explore various possibilities and discover the best allocation of resources for optimal production.

The parallel scheduling of activities in the phenotype ensures efficient resource usage and potentially reduces the overall production time by taking advantage of overlapping operations. However, it also demands careful consideration of resource availability and possible conflicts to maintain the feasibility of the manufacturing process.

### Avoidance of Occupied Machines

The concept of occupied machines comes into play through user input, where machine identification and corresponding intervals of occupancy are provided. By considering the start date of the schedule as timeslot 0, these intervals are then transformed into timeslots, forming a structured representation.

As activities are scheduled during the process of converting the genotype to phenotype, the availability of the machines is a crucial factor. For each activity, the correspondent machine's activity line is analyzed to identify feasible intervals where this new activity can be accommodated. The feasibility assessment takes into account the end date of the parent activity, which serves as the feasible start date, and searches for the first interval with a duration equal or greater than the activity being scheduled. If the activity lacks a parent, the feasible start timeslot defaults to 0.

This process can be observed through an example depicted in Figure 5.12. Suppose activity A is already scheduled on machine *m1*, and we are focusing on scheduling the subsequent activity, activity B, on machine *m2*. Activity B will be inserted in the first available slot following the end date of activity A.

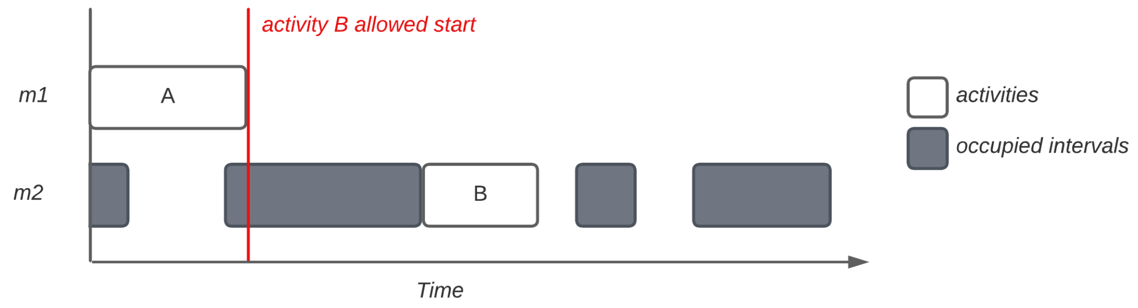


Figure 5.12: Example of scheduling considering a machine with occupied intervals.

### 5.4.4 Fitness Evaluation

Fitness evaluation is a crucial aspect of the GA as it represents the requirements the population should adapt to meet. It forms the basis for selection, and so it facilitates improvements. More accurately, it defines what improvement means [Eiben and Smith, 2015].

In this context, the goal is to minimize the fitness value, as lower values indicate better schedules. The fitness function is responsible for assigning a fitness score to each individual in the population based on specific criteria.

The fitness evaluation process includes the following components:

1. **Due Date Penalization**: If an activity finishes after its due date, the individual (schedule) is penalized. The penalty is proportional to the number of time slots that have passed since the due date, with a weight factor ($W_{DueDatePenalization}$) to adjust the penalty's impact on the overall fitness score.

2. **Machine Penalty**: The fitness function penalizes individuals that use more machines than necessary. Using additional machines can lead to inefficient resource usage. The penalty is based on the number of unique machines used in the schedule, multiplied by a weight factor ($W_{MachinePenalization}$) to adjust the penalty's influence. The penalty is purposefully maintained small in order to differentiate between two individuals who have the same end date schedule but employ different quantities of machines. This encourages the algorithm to find solutions that make more efficient use of machines while achieving the same end date, promoting optimal scheduling strategies.

3. **Interval Calculation**: To ensure that schedules are feasible, the fitness function calculates the interval covered by the schedule. This is the time span from the earliest start slot to the latest end slot for all activities in the schedule.

Given the fitness evaluation components previously described, the formula for the fitness evaluation can be summarized as follows:

$$\begin{aligned} Fitness = & \ (MaxEndSlot - MinStartSlot) \\ & + \sum TimeslotsFromDueDate_{activity} \cdot W_{DueDatePenalization} \\ & + MachinesUsed \cdot W_{MachinePenalization}, \end{aligned}$$

where:

- $MaxEndSlot$: The latest timeslot at which any activity finishes;
- $MinStartSlot$: The earliest timeslot at which any activity starts in the schedule;
- $\sum TimeslotsFromDueDate_{activity}$: The sum of timeslots that have passed since the established due dates for each activity that finishes after its due date;
- $MachinesUsed$: The total number of unique machines used in the schedule;

### 5.4.5 Genetic Operators

The primary objective of genetic operators is to generate new individuals from existing ones. In the corresponding phenotype space, this translates to the creation of new candidate solutions [Eiben and Smith, 2015].

To achieve this, genetic operators are classified into two types: Mutation and Crossover (also known as Recombination).

For this research, we opted to use the two-point crossover and a custom mutation operator. These specific operators were chosen to introduce diversity and explore different potential solutions in the Genetic Algorithms (GAs) search process.

#### Two-point Crossover

The two-point crossover is a popular genetic operator used in GAs for creating new individuals through recombination. It involves selecting two parent individuals from the population and randomly choosing two points along their chromosomes. The genetic information between these two points is then exchanged between the parents, producing two offsprings [Brabazon et al., 2016].

In the context of this work, the two-point crossover is applied by first randomly selecting a part from the individuals to be changed. Subsequently, two points are randomly selected within the activity list of those individuals. Since all individuals have the same set of parts to be scheduled, and consequently, the same list of activities to be processed, there are no issues in selecting these two points as the sequences are equal. Figure 5.13 illustrates an example of this crossover, where the interval between the two points in the parent individuals is modified, thereby creating two new individuals.

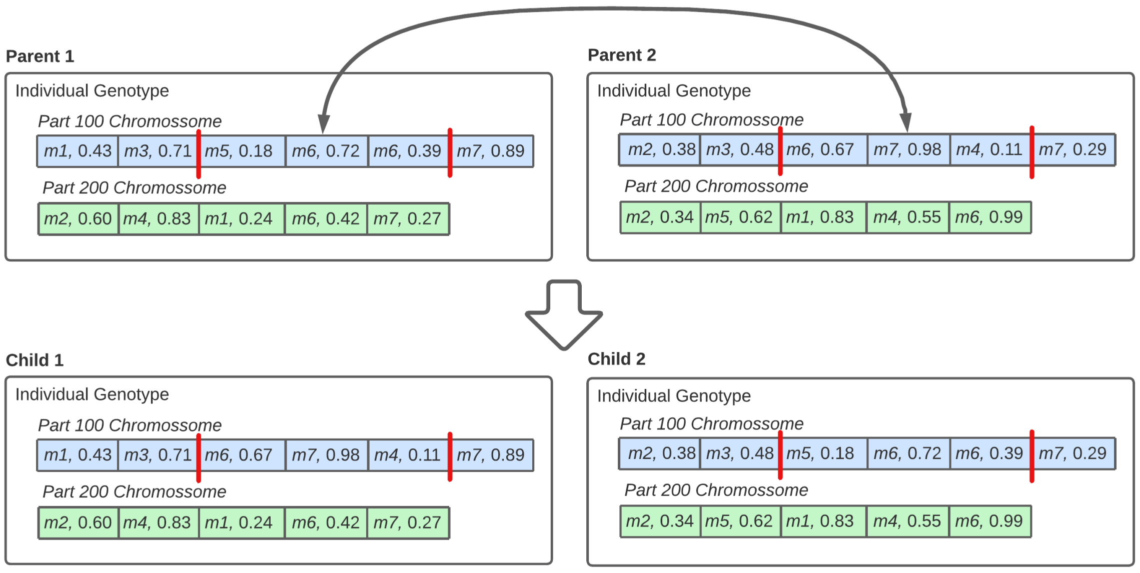

Parent 1
Individual Genotype
Part 100 Chromossome
m1, 0.43 | m3, 0.71 | m5, 0.18 | m6, 0.72 | m6, 0.39 | m7, 0.89
Part 200 Chromossome
m2, 0.60 | m4, 0.83 | m1, 0.24 | m6, 0.42 | m7, 0.27
Parent 2
Individual Genotype
Part 100 Chromossome
m2, 0.38 | m3, 0.48 | m6, 0.67 | m7, 0.98 | m4, 0.11 | m7, 0.29
Part 200 Chromossome
m2, 0.34 | m5, 0.62 | m1, 0.83 | m4, 0.55 | m6, 0.99
Child 1
Individual Genotype
Part 100 Chromossome
m1, 0.43 | m3, 0.71 | m6, 0.67 | m7, 0.98 | m4, 0.11 | m7, 0.89
Part 200 Chromossome
m2, 0.60 | m4, 0.83 | m1, 0.24 | m6, 0.42 | m7, 0.27
Child 2
Individual Genotype
Part 100 Chromossome
m2, 0.38 | m3, 0.48 | m5, 0.18 | m6, 0.72 | m6, 0.39 | m7, 0.29
Part 200 Chromossome
m2, 0.34 | m5, 0.62 | m1, 0.83 | m4, 0.55 | m6, 0.99

Figure 5.13: Example of the two-point crossover.

### Mutation operator

The mutation operator is essential in the GA as it keeps the search process going. In each iteration of the algorithm, mutation can find useful new solutions. On the other hand, if we rely solely on crossover to diversify the population, it will stop producing new solutions once all individuals become similar [Brabazon et al., 2016].

In our approach, the mutation process involves regenerating the resources associated with an activity. It is important to note that we cannot change the order of activities within a part chromosome.

When an individual has multiple feasible machines for an activity, we use a random process to determine the mutation. There is an equal probability (i.e., a 50% chance) of either changing the machine to which the activity is allocated or altering the priority of the activity in the machine queue. However, if there's only one feasible machine for the activity, we will focus on changing its priority.

Figure 5.14 depicts a basic example of the mutation process. As illustrated, part 200's sequence activity "Erosion" is mutated, resulting in a change in its assigned machine from *m6* to *m4*.

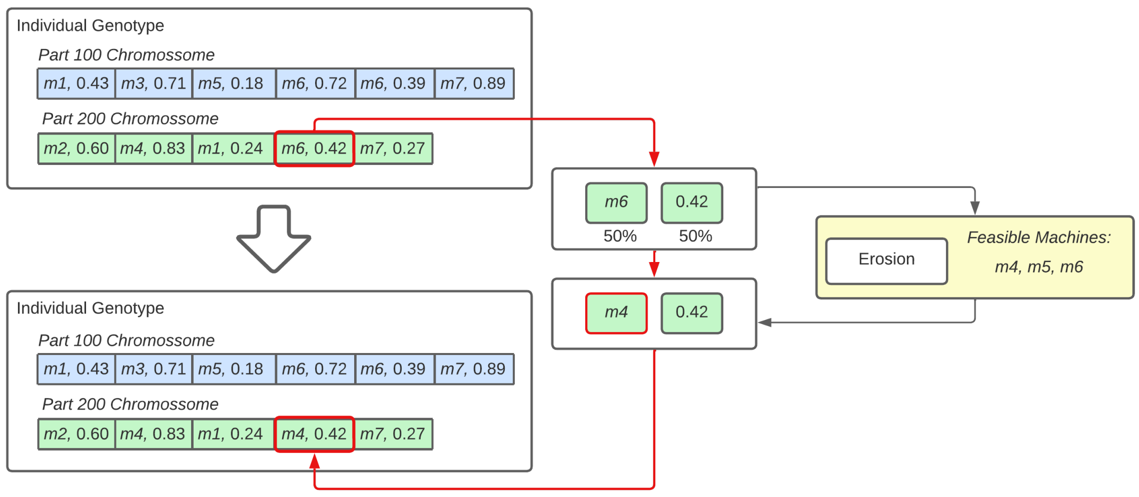


Figure 5.14: Example of the mutation operator.

### 5.4.6 Parent Selection

Parent selection is a technique inspired by natural selection, where individuals from a population are chosen to be parents for the next generation. They contribute their traits to create offspring, making the algorithm explore and exploit the search space effectively. The method used to choose parents determines the selection pressure, which affects how strongly higher-fitness individuals are favored. If the pressure is too low, good traits spread slowly, leading to inefficiency. If it is too high, the population may get stuck in a local optimum due to reduced genetic diversity. Finding the right balance is crucial for an effective optimization process [Brabazon et al., 2016].

In our approach, we employed a specific parent selection method known as "tournament selection". Tournament selection is a probabilistic which involves randomly selecting a group of individuals from the population, and the best-performing individual among them becomes a parent for the next generation [Brabazon et al., 2016].

For instance, if we set the tournament size as $k = 3$, three individuals are randomly chosen from the population, and the one with the highest fitness is selected for reproduction. As a result, the fittest individual in this small group becomes the tournament winner and a parent for the next generation. It is worth noting that the higher the value of $k$, the higher the selective pressure, as larger tournaments favor the selection of fitter individuals.

The tournament selection process is repeated $n$ times, where $n$ represents the population size. Each iteration selects a new parent, and this process continues until the required number of parents for the next generation is obtained. This strategy allows for individual diversity and competition, thereby contributing to the evolutionary process in genetic algorithms.

### 5.4.7 Elitism

In our approach, we adopted the Elitism strategy, which is a Fitness-Based replacement strategy that guarantees that the fittest members of the current population will always survive into the next generation [Brabazon et al., 2016].

The main purpose of elitism is to preserve the currently fittest individuals in the population. By preserving these top-performing individuals from one generation to the next, we ensure that their valuable traits and characteristics remain throughout evolution.

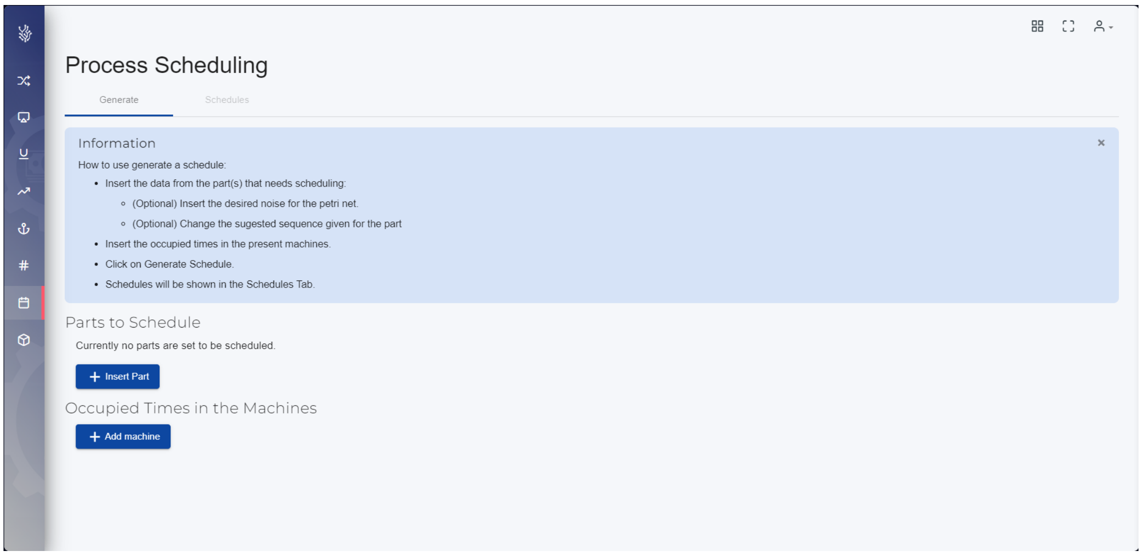
While Elitism primarily focuses on exploiting the solution space and concentrating on a local optimum, it also serves as a safety net in case that the algorithm cannot discover better solutions. By preserving the best individuals from each iteration, elitism ensures that the algorithm retains the most effective solutions discovered to date. This safeguard provides a valuable foundation in the search process, preventing the risk of losing the most promising solutions and sustaining progress even when other techniques, such as tournament selection, may not produce superior results.

The level of elitism can be modified using a percentage parameter. This parameter determines the proportion of the fittest individuals in the population that will be preserved for future generations. Changing this proportion influences the rate at which the population converges on the optimal solutions found, by permitting a flexible balance between exploration and exploitation in the evolutionary process.

## 5.5 Frontend Implementation and Workflow

In order to integrate the Process Scheduling module with the Prom4Prod project, a new dedicated page was added to the existing frontend. This additional page serves as the interface through which users can communicate and exchange information related to the parts to be scheduled. In this section, we provide an overview of the first stable version of this added module, showcasing the primary page as seen in Figure 5.15.

Within the frontend, users can initiate the scheduling process by clicking on "Insert Part", which opens a stepper form. In this form, users can input crucial data such as production deadlines, part, and mold specifications, as depicted in Figure 5.16. Additionally, the user has the option to modify the noise threshold of the Inductive Miner algorithm, offering more control over the generated Petri net and impacting the discovered sequence accordingly (Figure 5.17).



Process Scheduling

Information

How to use generate a schedule:

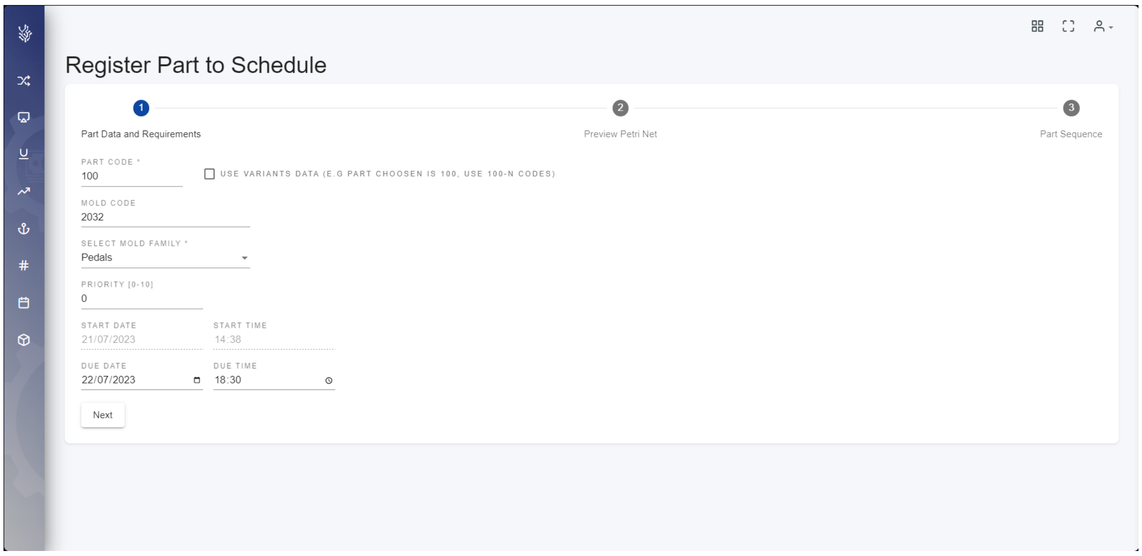
- Insert the data from the part(s) that needs scheduling:
  - (Optional) Insert the desired noise for the petri net.
  - (Optional) Change the sugested sequence given for the part
- Insert the occupied times in the present machines.
- Click on Generate Schedule.
- Schedules will be shown in the Schedules Tab.

Parts to Schedule

Currently no parts are set to be scheduled.

Occupied Times in the Machines

Figure 5.15: Initial Page of the Process Scheduling module.



Register Part to Schedule

| | |
|---|---|
| PART CODE * | 100 |
| USE VARIANTS DATA (E.G PART CHOOSEN IS 100, USE 100-N CODES) | ☐ |
| MOLD CODE | 2032 |
| SELECT MOLD FAMILY * | Pedals |
| PRIORITY [0-10] | 0 |
| START DATE | 21/07/2023 |
| START TIME | 14:38 |
| DUE DATE | 22/07/2023 |
| DUE TIME | 18:30 |

Figure 5.16: Data Input form of the frontend software module.

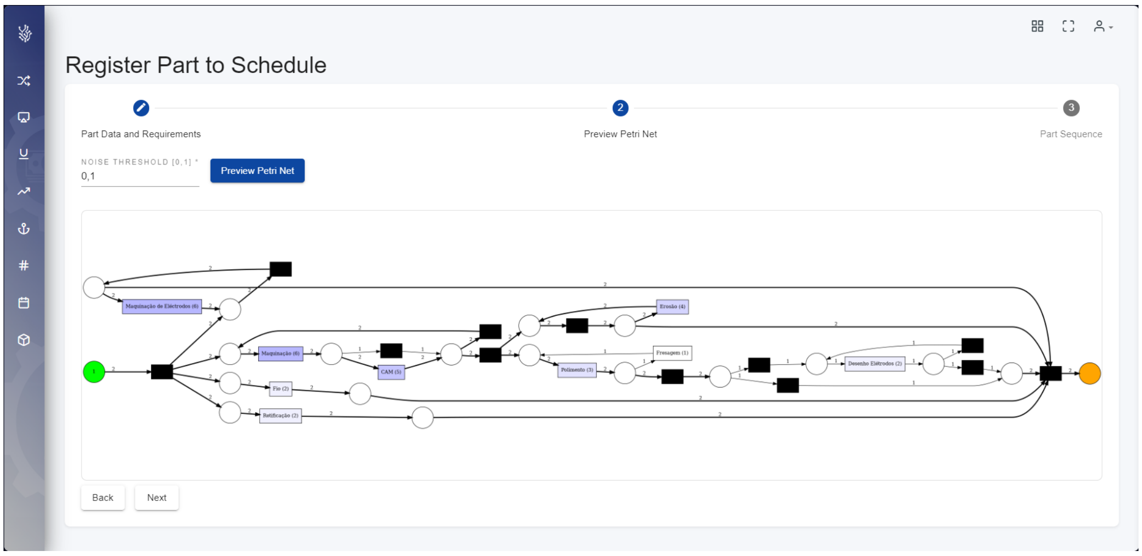
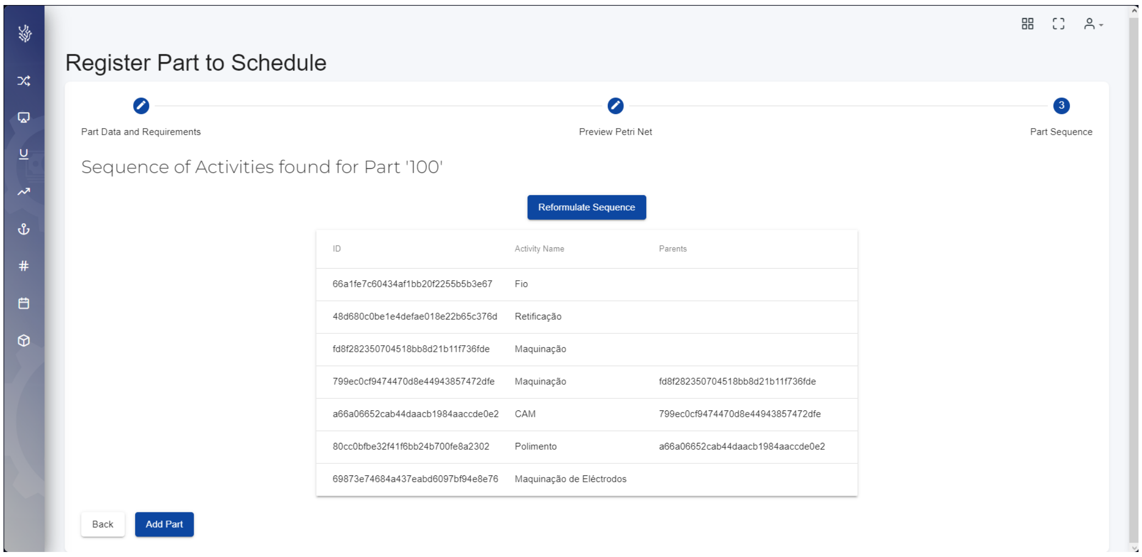


Figure 5.17: Generated Petri net with the noise threshold option.

Once the sequence is generated, the user can review it and make adjustments if needed, ensuring it aligns with the specific requirements of the part being scheduled (Figure 5.18). In future versions of the frontend, we aim to provide users with the ability to further manipulate the sequence, such as removing or reordering activities, granting even more manual control over the suggested sequence.

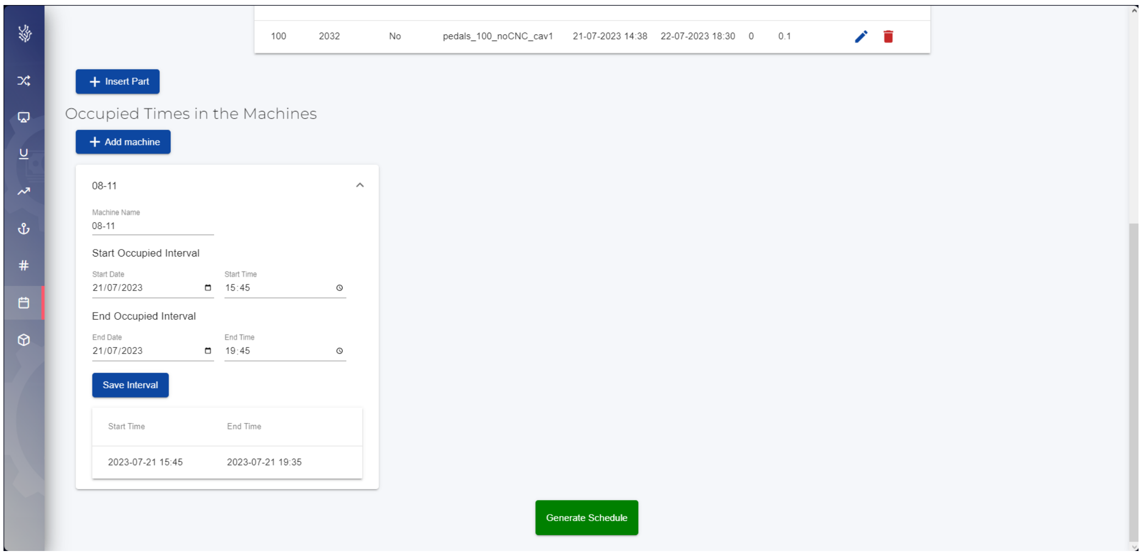


Sequence of Activities found for Part '100'

| ID | Activity Name | Parents |
|---|---|---|
| 66a1fe7c60434af1bb20f2255b5b3e67 | Fio | |
| 48d680c0be1e4defae018e22b65c376d | Retificação | |
| fd8f282350704518bb8d21b11f736fde | Maquinação | |
| 799ec0cf9474470d8e44943857472dfe | Maquinação | fd8f282350704518bb8d21b11f736fde |
| a66a06652cab44daacb1984aaccde0e2 | CAM | 799ec0cf9474470d8e44943857472dfe |
| 80cc0bfbe32f41f6bb24b700fe8a2302 | Polimento | a66a06652cab44daacb1984aaccde0e2 |
| 69873e74684a437eabd6097bf94e8e76 | Maquinação de Eléctrodos | |

Figure 5.18: Suggested sequence to be scheduled.

Additionally, users have the option to include resource availability by clicking on "Add Machines", as illustrated in Figure 5.19. This functionality allows users to specify occupied time intervals for the existing machines on the factory floor, providing the respective machine IDs. By doing so, these time intervals will be considered as reserved, influencing the scheduling process by preventing activities from occupying those locked timeslots during the specified intervals.



| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 100 | 2032 | No | pedals_100_noCNC_cav1 | 21-07-2023 14:38 | 22-07-2023 18:30 | 0 | 0.1 |

Occupied Times in the Machines

08-11

Machine Name: 08-11

Start Occupied Interval

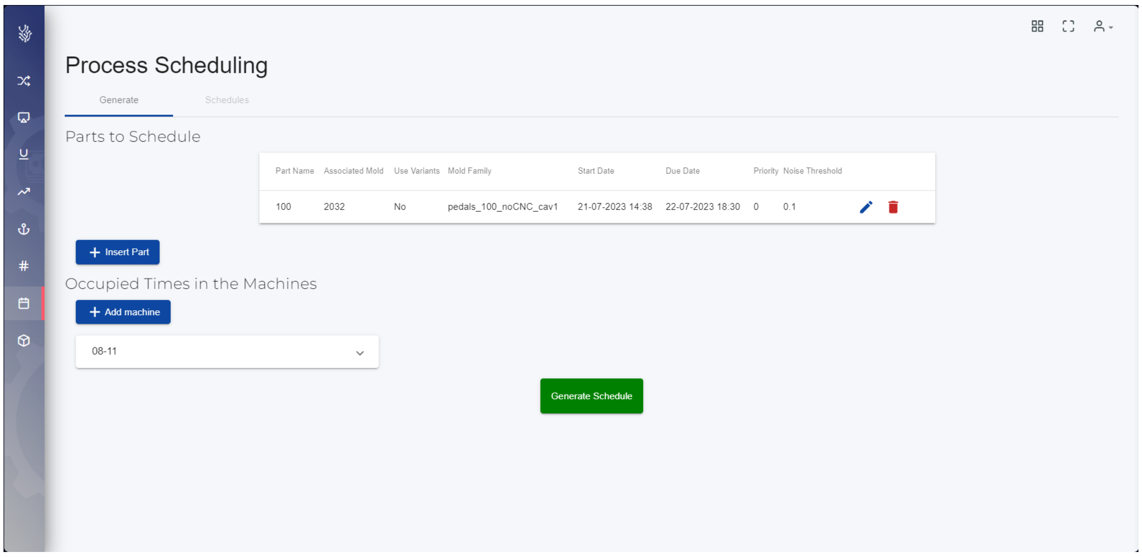
Start Date: 21/07/2023
Start Time: 15:45

End Occupied Interval

End Date: 21/07/2023
End Time: 19:45

| Start Time | End Time |
|---|---|
| 2023-07-21 15:45 | 2023-07-21 19:35 |

Figure 5.19: Set up of an occupied time interval.



Process Scheduling

Parts to Schedule

| Part Name | Associated Mold | Use Variants | Mold Family | Start Date | Due Date | Priority | Noise Threshold |
|---|---|---|---|---|---|---|---|
| 100 | 2032 | No | pedals_100_noCNC_cav1 | 21-07-2023 14:38 | 22-07-2023 18:30 | 0 | 0.1 |

Occupied Times in the Machines

08-11

Figure 5.20: Generate schedules button.

Finally, the user can initiate the scheduling process by clicking on the "Generate Schedules" button, as depicted in Figure 5.20. Once activated, the frontend transmits all the essential data to the backend, where the genetic algorithm is be executed three times, corresponding to each scenario (optimistic, realistic, and pessimistic).

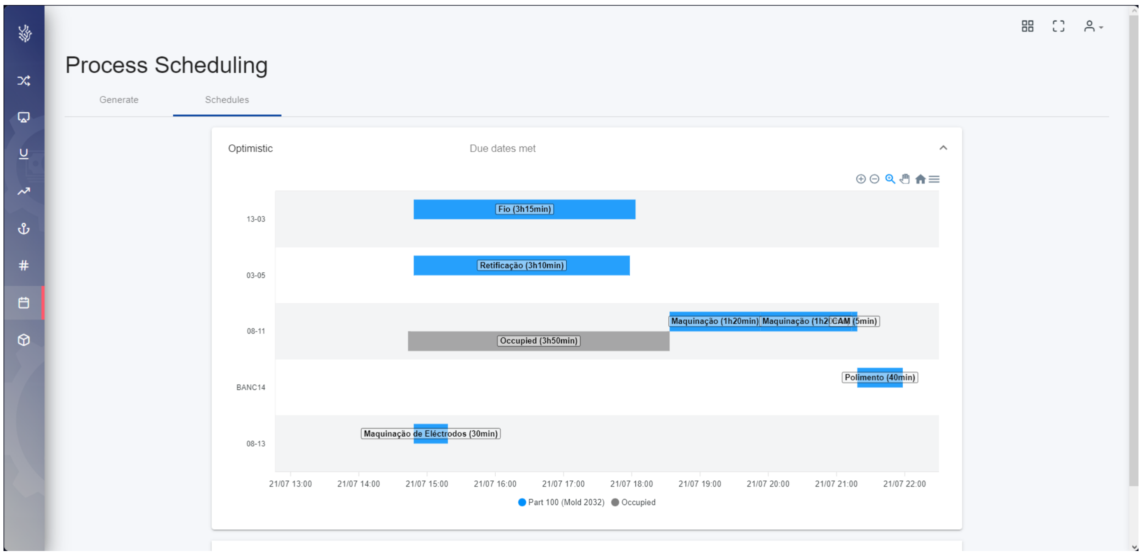


Figure 5.21: Result of the optimistic scenario.

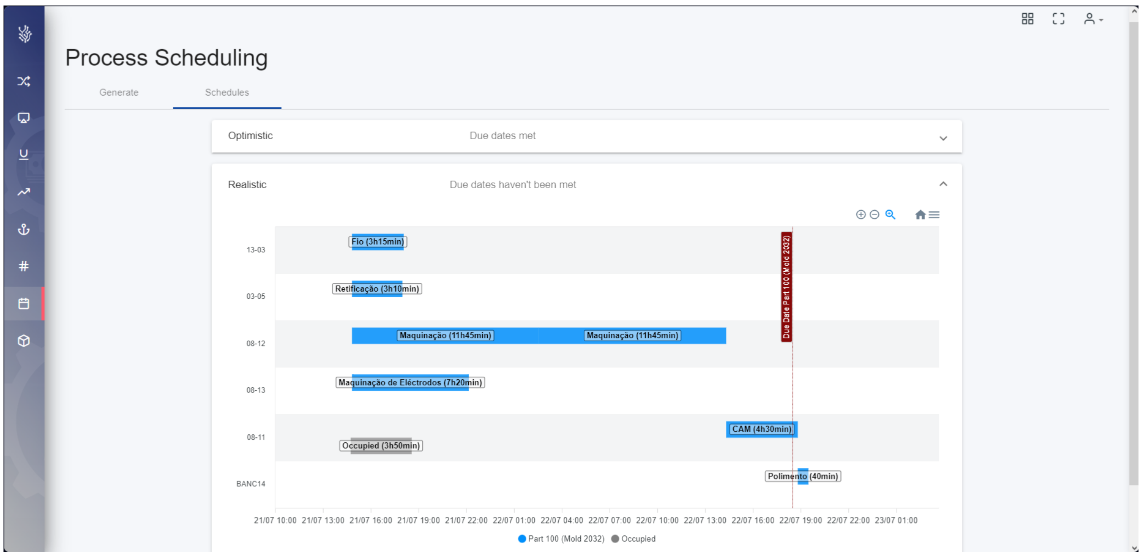


Figure 5.22: Result of the realistic scenario.

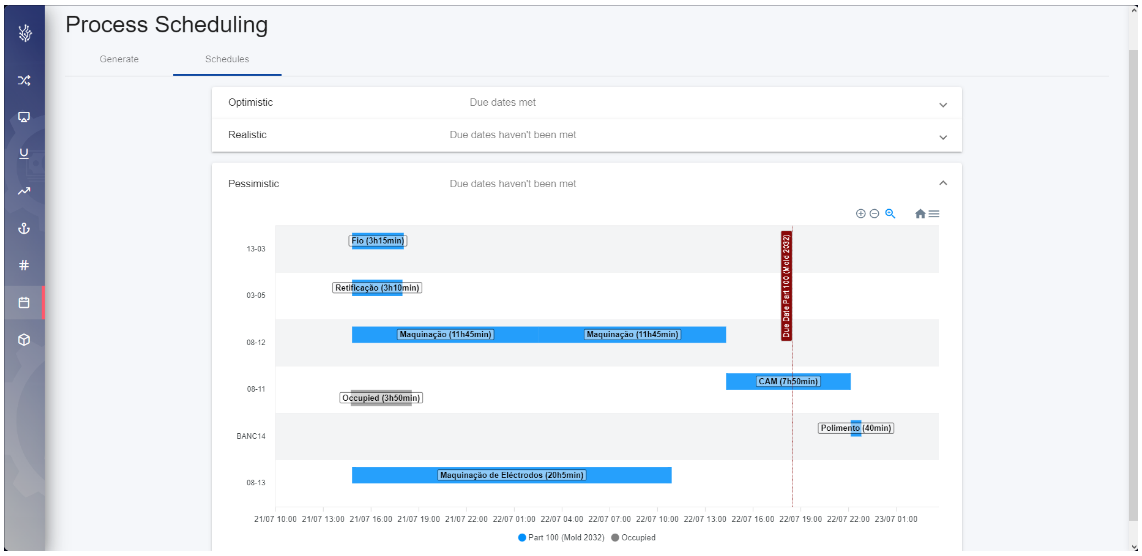


Figure 5.23: Result of the pessimistic scenario.

Figure 5.21, 5.22 and 5.23 display the completed output, which consists of three distinct Gantt charts, each representing a different scenario. These Gantt charts offer a visual representation of the allocated activities and whether the established due dates were met for each scenario. For instance, Figure 5.22 and 5.23 illustrate instances where the due dates were not met, indicating delays in the production process. Furthermore, the schedules in these figures clearly reflect any occupied time intervals specified for the machine with ID "08-11".

This output enables users to evaluate the effectiveness of the scheduling process under various conditions, thus enabling informed decision-making and production process optimization.

By developing this solution, we have successfully fulfilled the current requirements outlined in this document, specifically in Section 5.1. We have demonstrated its implementation by showcasing the frontend, providing factory managers and workers with a user-friendly approach to scheduling the production of required mold parts.

In the following chapter, we will evaluate the functionality and performance of this solution using real-world scenarios through testing and validation. Then, we will engage in an in-depth discussion of the overall work, including its limitations, thus further aligning our work with its original objectives.

# Chapter 6

# Results and Discussion

In this chapter, we discuss the effectiveness of our approach, showcasing its performance, and engage in a discussion about the overall study. We begin by validating the approach using real-world data and comparing the Process Mining (PM) output with the achieved schedules.

After completing the approach validation, our focus shifts to the comparative testing phase. This involves a direct comparison between the approach's schedule solution and an actual mold part produced on the factory floor. Finally, we delve into the performance testing, where we evaluate the system's performance under various conditions by varying the quantity of parts to be scheduled in different scenarios.

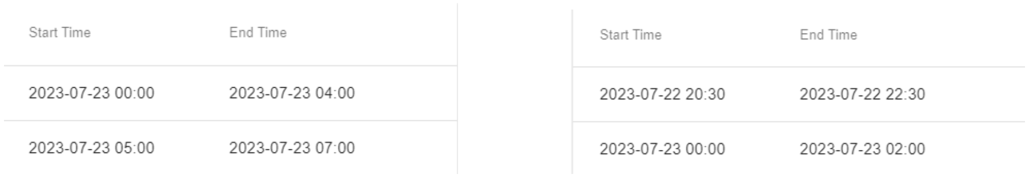
Throughout the discussion segment, we look into the results obtained from both the approach validation and performance assessment, including the outcomes of the comparative testing. We examine the strengths and limitations of this approach, highlighting its practical significance for the mold industry.

## 6.1 Approach Validation

In our validation testing, we intended to confirm the relation between the resultant schedules and the sequences found in the Petri net, while ensuring that these sequences are logically compatible.

To achieve this, we analyzed whether the schedules made use of feasible resources, avoided activity overlaps on the machines, and conformed to the already occupied timeslots. By confirming these criteria, we can confidently assert that our approach is logically sound.

The validation test described starts on the 22th of July 2023, at 20:30. During this test, two specific parts, Part 100 and Part 200, commonly referred to as a cavity and a bushing, respectively, were selected from the pedals mold family to be scheduled. This mold family is represented in the real dataset we possess and serves as the target for scheduling. As part of the evaluation process, we designed a scenario where two machines have occupied intervals to test the schedule's



| Start Time | End Time |
|---|---|
| 2023-07-23 00:00 | 2023-07-23 04:00 |
| 2023-07-23 05:00 | 2023-07-23 07:00 |

(a) Machine "07-07"

| Start Time | End Time |
|---|---|
| 2023-07-22 20:30 | 2023-07-22 22:30 |
| 2023-07-23 00:00 | 2023-07-23 02:00 |

(b) Machine "08-11"

Figure 6.1: Occupied intervals in the machines.

ability to adapt to previously allocated resources.

Throughout the entire validation process, we compared the obtained results with the event logs from the mold family and the Petri net derived through PM.

The occupied time intervals for machines "07-07" and "08-11" are shown in Figure 6.1. Machine "07-07" is initially free for 4 hours, then occupied for 4 hours, followed by a 1-hour free interval before becoming occupied again for an additional 2 hours. Machine "08-11" is occupied for the first 2 hours, then has a 1-hour and 30-minute free interval before becoming occupied for 2 more hours. These intervals serve as a crucial test to examine the schedule's capacity to efficiently allocate resources without conflicts with pre-existing schedules.

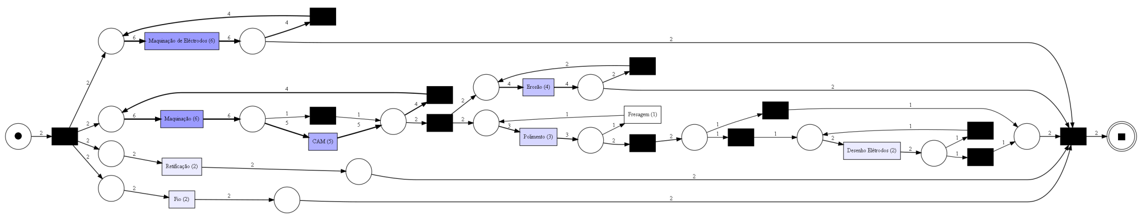


Figure 6.2: Petri net representing part 100 (cavity) in the "Pedals" mold family.

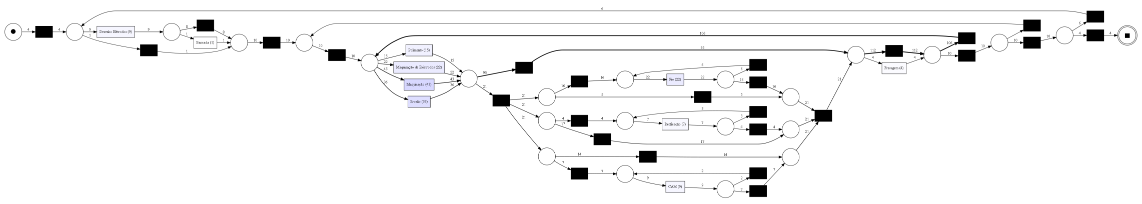


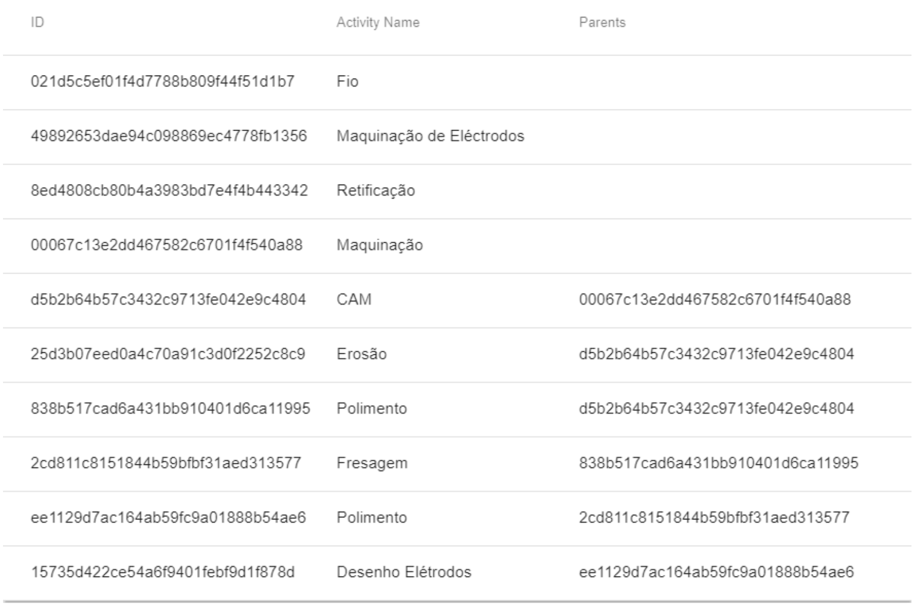
Figure 6.3: Petri net representing part 200 (bushing) in the "Pedals" mold Family.

The Petri net representing part 100 (cavity) process can be seen in Figure 6.2, which was created using the Inductive Miner Infrequent algorithm with a noise threshold of 0.1, allowing for a small degree of noise in the observed processes. Additionally, in Figure 6.3, we have a Petri net representing part 200 (bushing) process, which was generated using the same algorithm but with a noise threshold of 0, indicating no limitation on the noise seen in the processes. Both of these parts were scheduled for a fictional mold with ID 2300.

Table 6.1: Resources used by activities from Part 100 and Part 200.

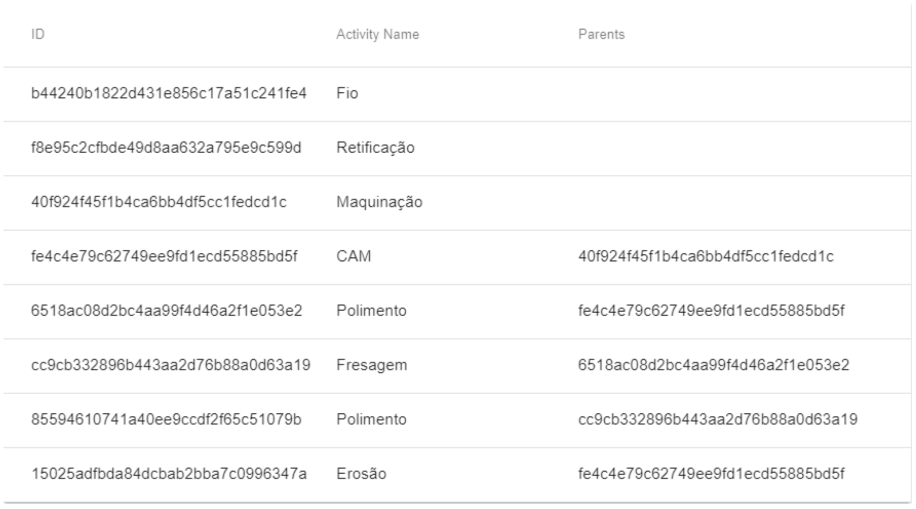
| **Activity** | **Machines** | |
|---|---|---|
| | Part 100 | Part 200 |
| CAM | 08-11 | 08-11 |
| CNC | 08-11, 08-12 | 08-10, 08-11, 08-12 |
| Countertop Work | - | BANC08 |
| Electrode Design | 08-11 | 08-11 |
| Electrode Machining | 08-13 | 08-13 |
| Erosion | 09-05, 09-06, 09-07 | 09-05, 09-06, 09-07 |
| Milling | 07-07 | 07-07 |
| Polishing | BANC14, BANC15 | BANC14, BANC15 |
| Rectification | 03-05 | 03-04, 03-05 |
| Wire | 13-03 | 13-03 |

Due to the extensive size of the event log, it is impractical to present it in its entirety. As an alternative, Table 6.1 showcases the resources used by each activity in Part 100 (cavity) and Part 200 (bushing) from the Pedals Mold Family.



| ID | Activity Name | Parents |
|---|---|---|
| 021d5c5ef01f4d7788b809f44f51d1b7 | Fio | |
| 49892653dae94c098869ec4778fb1356 | Maquinação de Eléctrodos | |
| 8ed4808cb80b4a3983bd7e4f4b443342 | Retificação | |
| 00067c13e2dd467582c6701f4f540a88 | Maquinação | |
| d5b2b64b57c3432c9713fe042e9c4804 | CAM | 00067c13e2dd467582c6701f4f540a88 |
| 25d3b07eed0a4c70a91c3d0f2252c8c9 | Erosão | d5b2b64b57c3432c9713fe042e9c4804 |
| 838b517cad6a431bb910401d6ca11995 | Polimento | d5b2b64b57c3432c9713fe042e9c4804 |
| 2cd811c8151844b59bfbf31aed313577 | Fresagem | 838b517cad6a431bb910401d6ca11995 |
| ee1129d7ac164ab59fc9a01888b54ae6 | Polimento | 2cd811c8151844b59bfbf31aed313577 |
| 15735d422ce54a6f9401febf9d1f878d | Desenho Elétrodos | ee1129d7ac164ab59fc9a01888b54ae6 |

Figure 6.4: Sequence generated for Part 100 (cavity).



| ID | Activity Name | Parents |
|---|---|---|
| b44240b1822d431e856c17a51c241fe4 | Fio | |
| f8e95c2cfbde49d8aa632a795e9c599d | Retificação | |
| 40f924f45f1b4ca6bb4df5cc1fedcd1c | Maquinação | |
| fe4c4e79c62749ee9fd1ecd55885bd5f | CAM | 40f924f45f1b4ca6bb4df5cc1fedcd1c |
| 6518ac08d2bc4aa99f4d46a2f1e053e2 | Polimento | fe4c4e79c62749ee9fd1ecd55885bd5f |
| cc9cb332896b443aa2d76b88a0d63a19 | Fresagem | 6518ac08d2bc4aa99f4d46a2f1e053e2 |
| 85594610741a40ee9ccdf2f65c51079b | Polimento | cc9cb332896b443aa2d76b88a0d63a19 |
| 15025adfbda84dcbab2bba7c0996347a | Erosão | fe4c4e79c62749ee9fd1ecd55885bd5f |

Figure 6.5: Sequence generated for Part 200 (bushing).

Figures 6.4 and 6.5 illustrate the sequences generated by the recursive algorithm. To establish a dependency between activities, a unique ID is assigned to each activity.

Figures 6.2 and 6.3 display the corresponding Petri nets derived from the sequences, providing better clarity and understanding. Through careful analysis, we can observe that these sequences can be originally constructed from the Petri nets extracted using the Inductive Miner Infrequent algorithm.

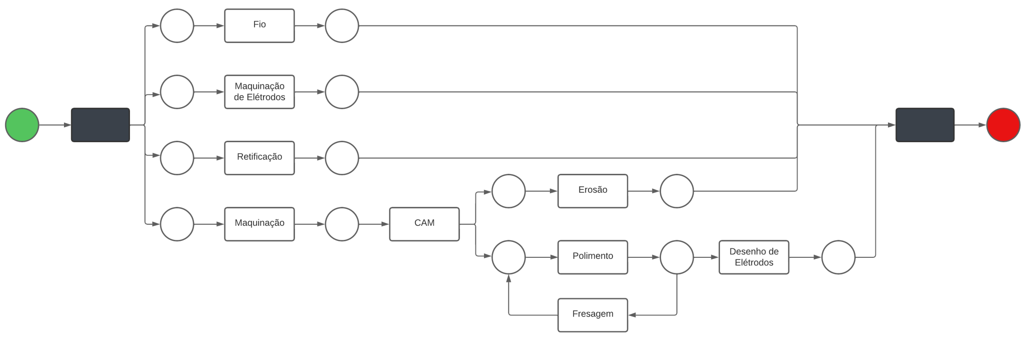


Figure 6.6: Sequence extracted for part 100 (cavity) in form of a Petri net.

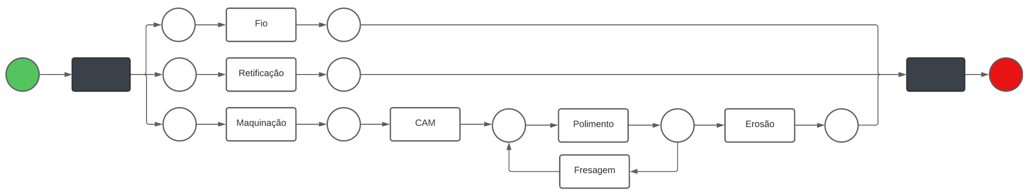


Figure 6.7: Sequence generated for part 200 (bushing) in form of a Petri net.

With the sequence validated, our focus shifts to validating the schedule. This step involves verifying whether the allocated resources are feasible, ensuring that the schedules match the dependencies identified in the sequence, and confirming that the activities do not overlap with each other or with occupied intervals.

By running the GA, we obtain three schedules for the three scenarios, represented in Figures 6.8, 6.9, and 6.10, respectively. By comparing with the sequences and with the feasible machines in Table 6.1, we can see that the schedules respect the previous stated requirements.

It is worth noting that due to the limitations of the library used for the Gantt charts, the visualization might be challenging. However, a useful feature is provided, whereupon hovering the mouse over each activity, we can view the exact start and end times for better insight into the scheduling details, making it possible to validate this information.

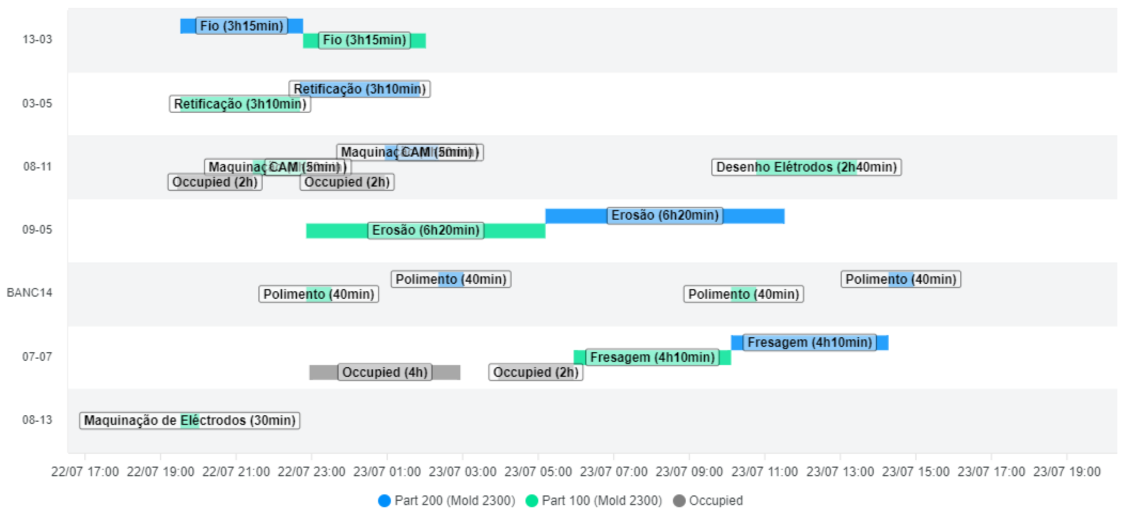


Figure 6.8: Validation of the optimistic schedule.

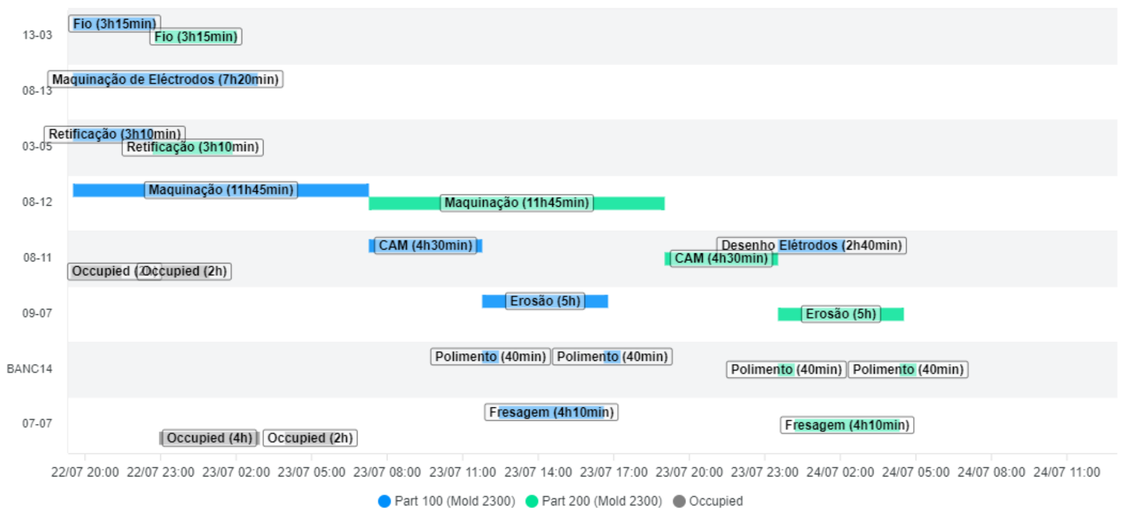


Figure 6.9: Validation of the realistic schedule.

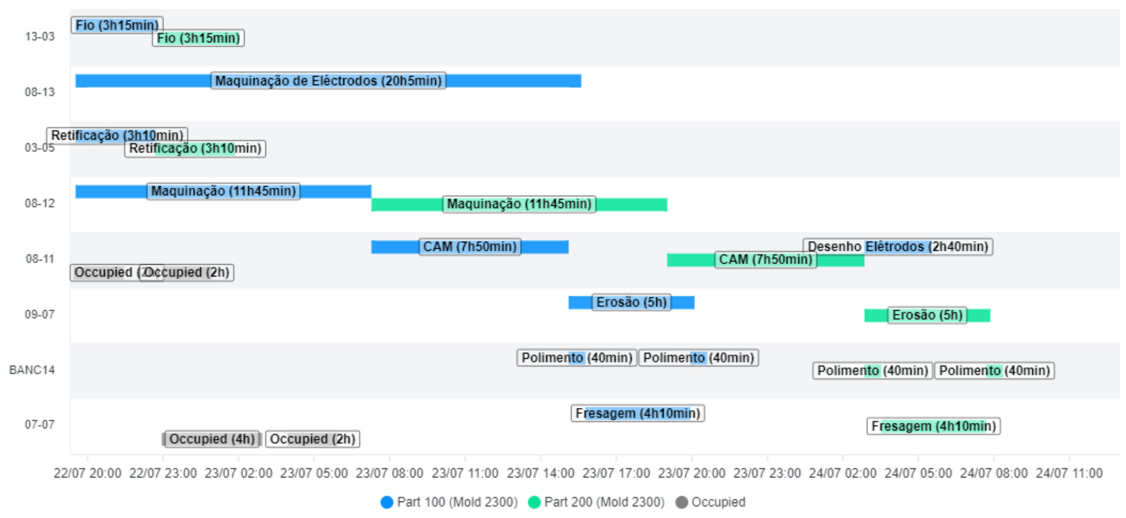


Figure 6.10: Validation of the pessimistic schedule.

With the validation test completed, we can now move to the comparative test.

## 6.2 Comparative Test

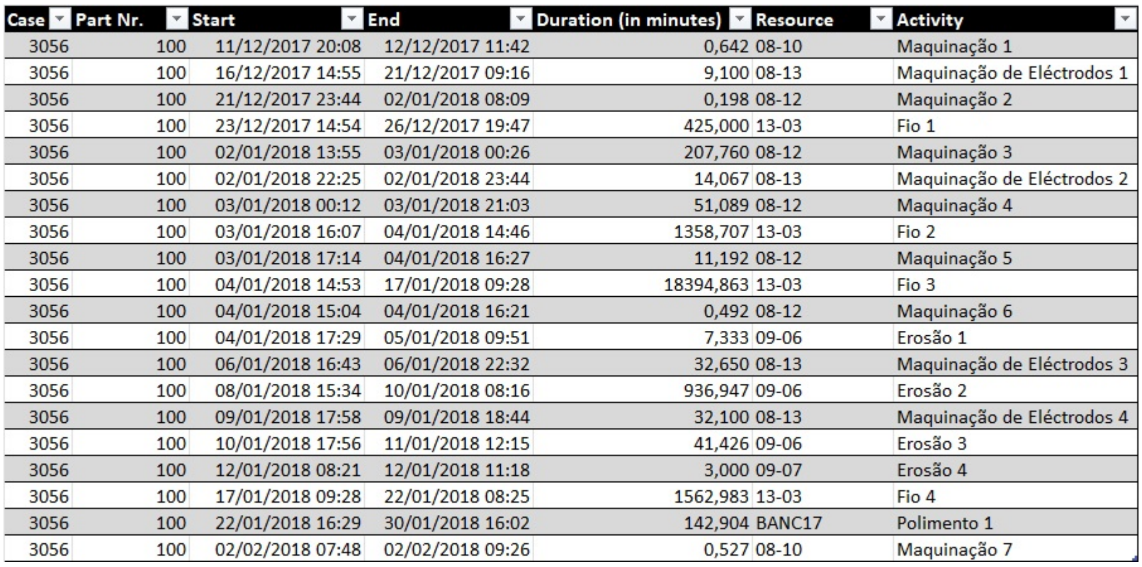
In conducting the comparative tests, we devised two distinct experiments. The initial experiment replicated the actual process involving a mold on the factory floor, in particular, we chose to isolate the production of Part 100, which corresponds to a cavity within Mold 3056, a mold from the coffee machine mold family. This involved employing the same resources and timeframes as those used in the real-life scenario, with the primary objective of analyzing the feasibility of executing the same process at an earlier time. This assessment aimed to identify any potential scheduling errors or human oversights that might have led to suboptimal timelines.

For the second experiment, we adopted an alternative approach by leveraging insights from the broader coffee machine mold family. Knowing the durations and machines used in other production processes for the Part 100, we aimed to determine if using other feasible resources could accelerate the production process for Part 100 of the Mold 3056.

### Testing with the actual process

To conduct this experiment, we began by extracting all relevant logs associated with the production of Part 100 from the database. Following this data extraction, we proceeded to adjust the activity names to ensure non-repetition and generate a Petri net with the Inductive Miner algorithm. This resulted in a Petri net configuration with straightforward relationships between 20 activities. Our intention was to replicate the exact process that occurred in the factory. These activities, their respective times and resources can be observed in Figure 6.11.

Following this initial extraction, we used the start time (11th of December 2017, at 20:08) and end time (2nd of February 2018, at 09:26) of the part 100 production process. This information was then employed to retrieve all event logs within that specific time frame from the database. By doing so, we were able to simulate the activities performed on the factory floor during that period. We subsequently proceeded with the data cleaning process, involving the concatenation of logs, a procedure elaborated upon in Section 5.3.1. These time intervals extracted from the event logs were then transformed into timeslots, establishing them as occupied intervals for further analysis.



| Case | Part Nr. | Start | End | Duration (in minutes) | Resource | Activity |
|---|---|---|---|---|---|---|
| 3056 | 100 | 11/12/2017 20:08 | 12/12/2017 11:42 | 0,642 | 08-10 | Maquinação 1 |
| 3056 | 100 | 16/12/2017 14:55 | 21/12/2017 09:16 | 9,100 | 08-13 | Maquinação de Eléctrodos 1 |
| 3056 | 100 | 21/12/2017 23:44 | 02/01/2018 08:09 | 0,198 | 08-12 | Maquinação 2 |
| 3056 | 100 | 23/12/2017 14:54 | 26/12/2017 19:47 | 425,000 | 13-03 | Fio 1 |
| 3056 | 100 | 02/01/2018 13:55 | 03/01/2018 00:26 | 207,760 | 08-12 | Maquinação 3 |
| 3056 | 100 | 02/01/2018 22:25 | 02/01/2018 23:44 | 14,067 | 08-13 | Maquinação de Eléctrodos 2 |
| 3056 | 100 | 03/01/2018 00:12 | 03/01/2018 21:03 | 51,089 | 08-12 | Maquinação 4 |
| 3056 | 100 | 03/01/2018 16:07 | 04/01/2018 14:46 | 1358,707 | 13-03 | Fio 2 |
| 3056 | 100 | 03/01/2018 17:14 | 04/01/2018 16:27 | 11,192 | 08-12 | Maquinação 5 |
| 3056 | 100 | 04/01/2018 14:53 | 17/01/2018 09:28 | 18394,863 | 13-03 | Fio 3 |
| 3056 | 100 | 04/01/2018 15:04 | 04/01/2018 16:21 | 0,492 | 08-12 | Maquinação 6 |
| 3056 | 100 | 04/01/2018 17:29 | 05/01/2018 09:51 | 7,333 | 09-06 | Erosão 1 |
| 3056 | 100 | 06/01/2018 16:43 | 06/01/2018 22:32 | 32,650 | 08-13 | Maquinação de Eléctrodos 3 |
| 3056 | 100 | 08/01/2018 15:34 | 10/01/2018 08:16 | 936,947 | 09-06 | Erosão 2 |
| 3056 | 100 | 09/01/2018 17:58 | 09/01/2018 18:44 | 32,100 | 08-13 | Maquinação de Eléctrodos 4 |
| 3056 | 100 | 10/01/2018 17:56 | 11/01/2018 12:15 | 41,426 | 09-06 | Erosão 3 |
| 3056 | 100 | 12/01/2018 08:21 | 12/01/2018 11:18 | 3,000 | 09-07 | Erosão 4 |
| 3056 | 100 | 17/01/2018 09:28 | 22/01/2018 08:25 | 1562,983 | 13-03 | Fio 4 |
| 3056 | 100 | 22/01/2018 16:29 | 30/01/2018 16:02 | 142,904 | BANC17 | Polimento 1 |
| 3056 | 100 | 02/02/2018 07:48 | 02/02/2018 09:26 | 0,527 | 08-10 | Maquinação 7 |

Figure 6.11: Activity data for part 100 of Mold 3056.

Given that we are examining a single process workflow without variations in resource combinations or diversity, both the sequence extraction and the GA operate deterministically. As there are no resource combinations or variations in this context, all individuals within the GA are identical, resulting in a static output, which can be observed in Figure 6.12. It can be observed the overall factory floor activity; in orange we can see the activities corresponding to the part that we are comparing.

Considering the complexity of understanding the suggested process times for Part 100, the visualization can be improved by hiding the occupied slots. Figure 6.13 illustrates the activities associated with Part 100 for a clearer perspective.

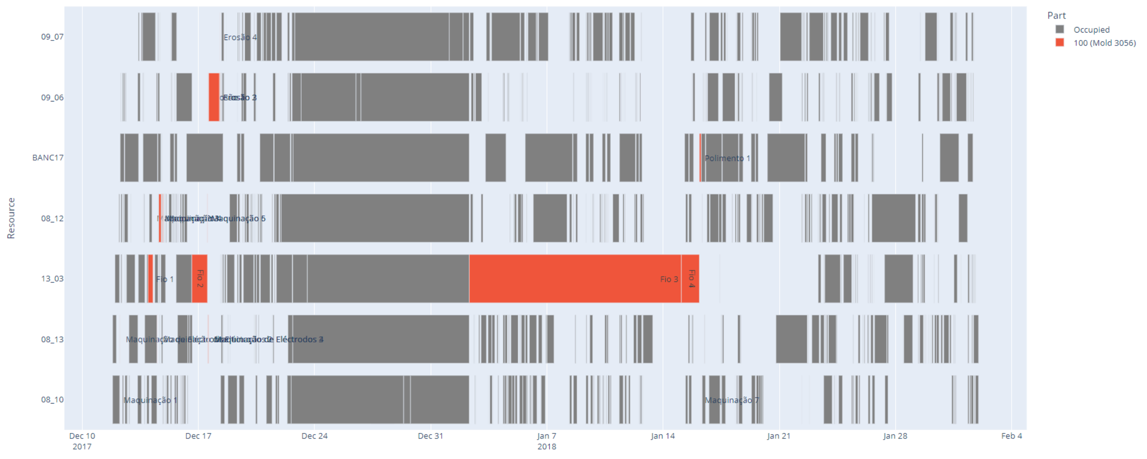


Figure 6.12: Big picture of the suggested schedule.

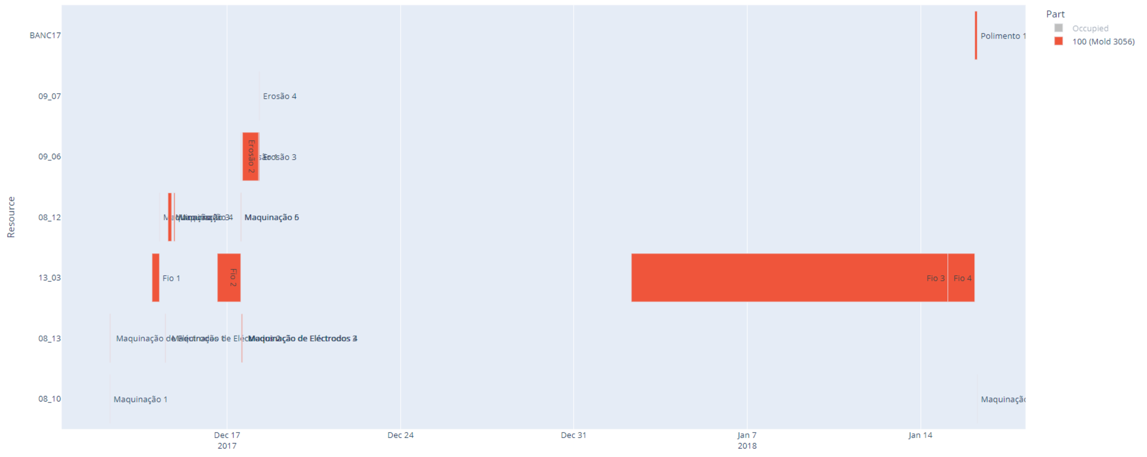
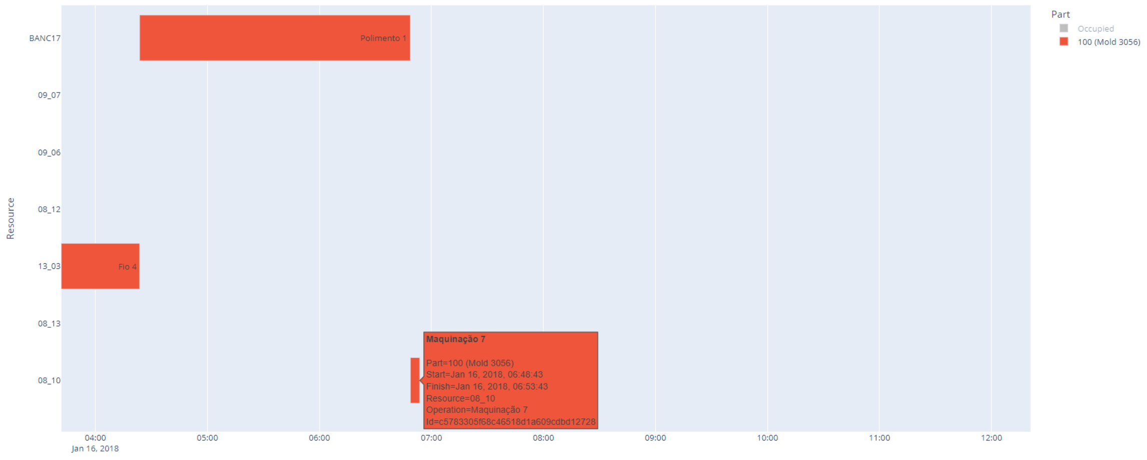


Figure 6.13: Suggested schedule for part 100.



```
Maquinação 7

Part=100 (Mold 3056)
Start=Jan 16, 2018, 06:48:43
Finish=Jan 16, 2018, 06:53:43
Resource=08_10
Operation=Maquinação 7
Id=c5783305f68c46518d1a609cdbd12728
```

Figure 6.14: Suggested schedule end date.

In Figure 6.14, the schedule is zoomed in on the final activity "Maquinação 7", which is predicted to conclude on the 16th of January 2018, at 06:53 AM. This prediction anticipates completion by 17 days compared to the actual process. The actual production spanned 52 days, 13 hours, 18 minutes, and 11 seconds, while the proposed schedule spanned 35 days and 25 minutes. Thus, observing that the suggested schedule achieved a 32.6% improvement over the real production time in this case study.

**Testing with different resources**

For our second experiment, we used the same data extracted in the previous experiment. However, instead of restricting the resources based solely on those employed in the production of Part 100 for Mold 3056, we broadened the scope. This time, we collected event logs associated with the production of Part 100 across various molds within the coffee machine mold family. By incorporating this data, we aimed to cover a more diverse range of available resources and their associated timeframes. This approach allowed us to explore alternative machinery options and potential duration adjustments, thus providing opportunities to optimize the Part 100 manufacturing process.

These event logs provided useful information, particularly regarding the feasibility of using different machines and the corresponding time durations for part 100 production. This resource extraction is documented in Section 5.3.1.

Given the range of available resources in comparison to the first experiment, where each activity was tied to a specific machine, the role of the GA becomes crucial in generating an optimal outcome.

After conducting 30 independent runs for each scenario (i.e., the optimistic, realistic, and pessimistic scenarios), we consistently observed the same end time across all runs within each scenario. However, there was evident variability in the usage of resources. This variability comes from the fact that, while certain machines may present faster processing times, their efficiency becomes restricted by subsequent intervals already occupied.

Figure 6.15 showcases a possible outcome for the optimistic schedule using different machines than in the real process. In Figure 6.16, it is observed the schedule of Part 100 with the occupied machines hidden, allowing a better visualization of the process. A comparison with Figure 6.17 reveals shorter activity durations, as we are using the most favorable estimated duration for each activity.

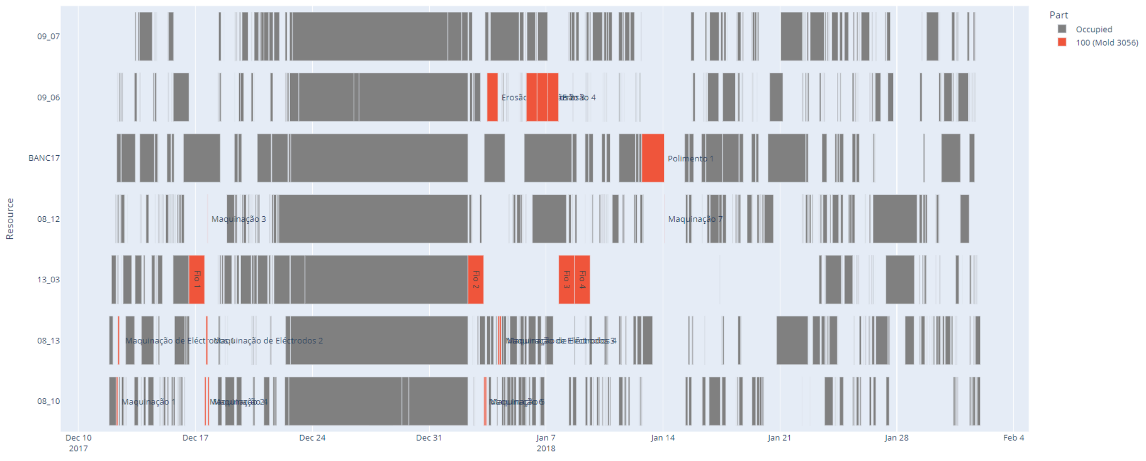


Figure 6.15: Big picture of the suggested optimistic schedule.

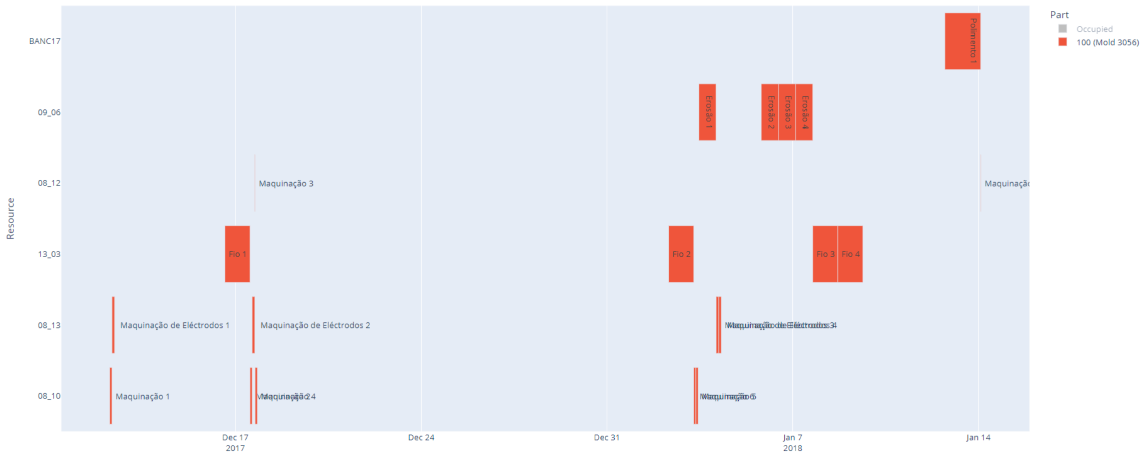


Figure 6.16: Suggested optimistic schedule.

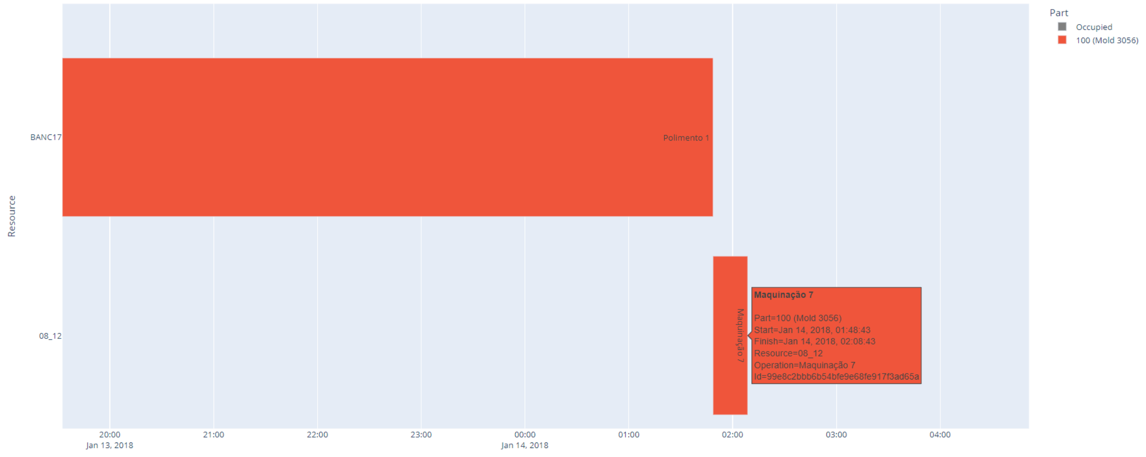


Figure 6.17: Suggested optimistic schedule end date.

In Figure 6.17, the schedule's end date is displayed. These results are better than those of the first experiment, because they result in a schedule concluding on the 14th of January 2018, at 02:08 AM, which is 2 days, 4 hours, and 45 minutes earlier than the first experiment's conclusion (equivalent to a performance increase of approximately 6.3%). Moreover, this results in a substantial advancement of 19 days, 7 hours, and 18 minutes compared to the actual end date, corresponding to an increase of approximately 37.5%.

Turning to the realistic and pessimistic scenarios, the outcomes were not as favorable. In contrast to the optimistic scenario, where the schedule concluded ahead of the actual end date, the results for both the realistic and pessimistic scenarios revealed that the schedule extended beyond the real end date. This implies that the durations estimated for the actual process are shorter than the predicted durations for the realistic and pessimistic scenarios.

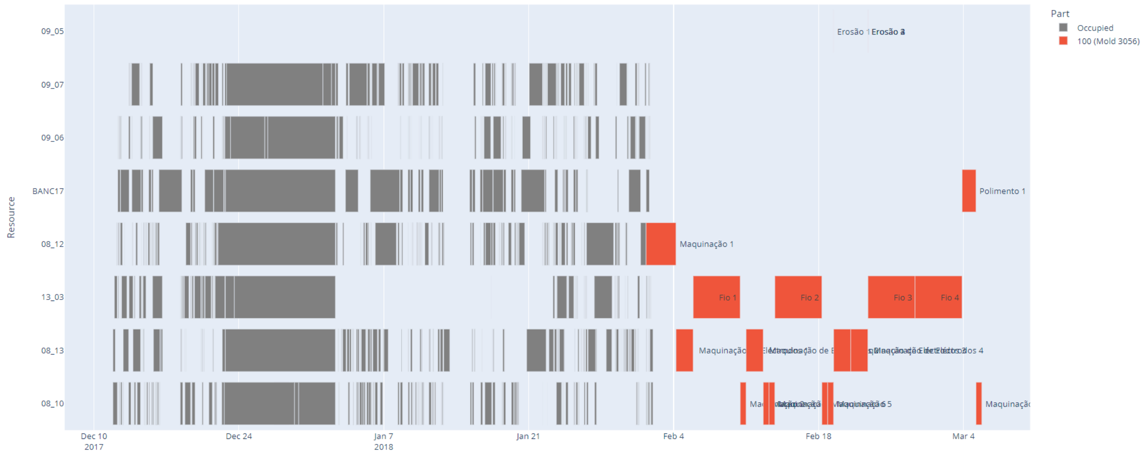


Figure 6.18: Suggested realistic schedule.

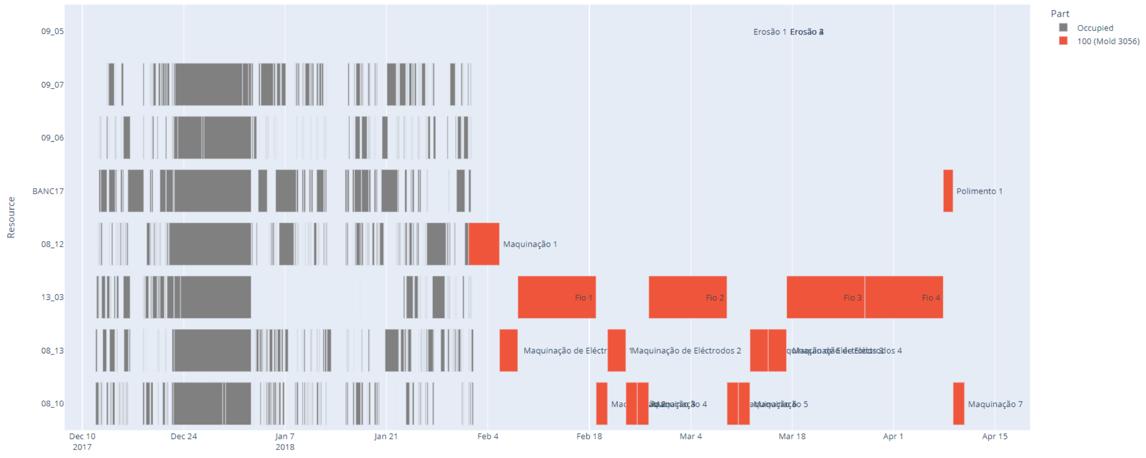


Figure 6.19: Suggested pessimistic schedule.

In analyzing the results depicted in Figures 6.18 and 6.19, it becomes evident that the outcomes for the realistic and pessimistic scenarios were influenced by inherent limitations beyond the scope of the GA.

In both the optimistic and pessimistic scenarios, the unfeasibility of executing the schedules within the specified timeframes is mainly due to resource occupancy constraints on the factory floor. In essence, the GA operated within the constraints of these real-world resource limitations, which significantly impacted the scheduling and operational efficiency of the production processes.

For instance, in the realistic scenario, the schedule ended on 5th of March 2018 at 18:33, resulting in a performance decrease of approximately 59.7% compared to the actual end date. Similarly, in the pessimistic scenario, the schedule ended on 10th of April 2018 at 18:58, corresponding to a performance decrease of approximately 128.2% compared to the actual end date. These outcomes underscore the significant impact of resource availability on the scheduling and efficiency of production processes.

In conclusion, the comparative tests show the potential of this scheduling approach to enhance production efficiency. In the context of the two experiments, the results offer distinct perspectives. The first experiment showcased the potential for an improved scheduling, as the scheduled scenario featured an earlier end date.

However, the second experiment, with its multiple scenarios, introduced a different perspective. While the optimistic scenario proposed a shorter schedule, even surpassing the first experiment's performance, the realistic and pessimistic scenarios presented a different challenge. These scenarios illustrated schedules that extended beyond the actual end date, highlighting the practical challenges associated with resource limitations and real-world manufacturing constraints.

This discrepancy can be attributed to the fact that the actual durations of activities in the production process were consistently shorter than the average durations captured in the limited dataset from the molds family. It is essential to note that the molds family dataset comprises only three molds, representing three distinct production processes of part 100. This limited dataset influenced the predicted durations, highlighting the need for a more extensive and diverse dataset to achieve accurate scheduling outcomes.

## 6.3 Performance Testing

For the performance testing of our approach, we designed five distinct problems to assess its behavior and efficiency when dealing with increased complexity. The focus of this testing was on the approach itself, rather than the entire architecture. Therefore, we concentrated on executing only the backend code and, as a result, the occupied time intervals were inputted in time slots using a default value of 5 minutes instead of real data. The frontend would later convert these dates to corresponding time slots.

The occupied time intervals for three machines, namely "07-07", "08-11", and "08-13", were defined as follows:

- **Machine "07-07":** [(70, 80), (120, 170)]
- **Machine "08-11":** [(10, 30), (50, 75)]

- **Machine "08-13":** [(0, 15), (100, 110)]

To determine the approach's performance, we constructed five different problem sets:

- **Problem 1:** Involves scheduling 1 part;
- **Problem 2:** Involves scheduling 5 parts;
- **Problem 3:** Involves scheduling 10 parts;
- **Problem 4:** Involves scheduling 15 parts;
- **Problem 5:** Involves scheduling 20 parts;

By varying the number of parts to be scheduled and introducing different occupied timeslots for the machines, we can observe how the approach handles increasing complexity. This performance testing provides valuable insights into its scalability and effectiveness in dealing with more demanding scenarios.

After conducting a series of preliminary experiments, we explored various combinations of parameterizations using the Optuna Framework[1], a Python library for hyperparameter optimization. Here are the results obtained from these experiments:

- **Number of Generations:** 62
- **Population Size:** 172
- **Probability of Crossover:** 0.5109
- **Probability of Mutation:** 0.4195
- **K-way tournament selection:** 5
- **Elitism Percentage:** 0.0265

To ensure the robustness of the results and eliminate any potential patterns, each problem was executed 30 times using different seeds.

Table 6.2 presents the average duration in seconds taken by the algorithm to solve the predefined scheduling problems with different numbers of parts to be scheduled. Additionally, the table includes the corresponding deviations, making it possible to understand the variability of the algorithm's performance across these scenarios. The results seem to show that the duration grows linearly with the number of parts.

The specifications of the machine used for conducting these tests are provided in Table 6.3.

[1]`https://optuna.org`

Moreover, the algorithm demonstrated a good performance, even when dealing with problems consisting of 20 parts, with an average duration of approximately 140 seconds.

Table 6.2: Problem durations.

| Parts to be Scheduled | Average Durations (Seconds) | Deviation |
|---|---|---|
| 1 | 3.529 | 0.139 |
| 5 | 32.979 | 1.325 |
| 10 | 46.626 | 1.968 |
| 15 | 101.627 | 2.17 |
| 20 | 140.321 | 7.08 |

Table 6.3: PC Hardware Specifications used for testing.

| Component | Specification |
|---|---|
| CPU | AMD Ryzen 7 5800H |
| GPU | NVIDIA GeForce RTX 3060 Laptop |
| RAM | 16 GB DDR4 |
| Storage | 500 GB NVMe SSD |
| Operative System | Windows 11 |

## 6.4 Discussion

This work presents an adaptation ofChoueiri and Santos [2021] approach to the mold industry. While Choueiri and Santos' work primarily dealt with scheduling identical products, our adaptation addresses the unique challenges posed by the mold industry, where each mold is distinct.

Therefore, by leveraging the concept of mold families and using various multiple filtering criteria such as the number of cavities, weight, cost, and parts used, we are able to identify similar molds. This allows us to analyze the production processes of these molds and extract valuable insights from the corresponding Petri net representations. Making it then possible to generate a schedule with the data acquired using a Genetic Algorithm (GA).

One of the most significant advantages of this approach is its reliance on real data from actual past processes for scheduling, rather than being based solely on engineers' assumptions about the process. This data-driven approach, facilitated by PM, ensures that the scheduling decisions are grounded in actual observed behavior, making the algorithm more accurate and capable of addressing real-world complexities of the processes.

The mold industry has traditionally relied on manual *ad hoc* planning, where individuals with extensive business and technical knowledge create schedules for use. However, by automating this process, scheduling can be performed in real-time, and the factory floor can be prepared with prior notice, leading to increased efficiency and responsiveness.

Drawing upon the fundamental idea of Choueiri and Santos, the implementation of three scenarios, optimistic, realistic, and pessimistic, offers more flexibility in decision-making within the factory. For example, by considering the pessimistic scenario, clients can be informed about a due date that may not be achievable beforehand, enabling better communication and managing expectations. On the other hand, adopting an optimistic scenario allows the factory to proactively manage raw materials used in mold parts, even contacting suppliers before exhausting the last resources, further streamlining operations.

Overall, the automation of scheduling and the use of scenario-based decision-making provide the mold industry with valuable tools to optimize production processes, enhance client communication, and effectively manage resources. These advancements can significantly contribute to increased productivity and operational excellence within the factory.

## 6.5 Limitations

One of the primary limitations of this approach is its lack of generality for other industries, or even within different companies in the mold industry. Since the processes involved are factory-specific, it becomes challenging to directly apply the same scheduling solution to other contexts without extensive customization.

While leveraging PM to base scheduling decisions on past data is advantageous, it also introduces certain limitations. For instance, the approach may struggle to detect new resources that have been introduced on the factory floor or processes (e.g., specific mold parts) that were not previously created regularly. This scarcity of event logs related to such newly introduced elements may result in insufficient knowledge about the resources capable of executing certain activities. Consequently, the metrics derived from limited data may not be precise enough to generate a fully reliable schedule.

Overfitting is a critical concern in the context of having low data and relying on historical information for scheduling decisions.

In the case of low data, the algorithm may have fewer process examples to extract meaningful sequences from. With limited event logs and historical records, the algorithm might excessively rely on these few data points, capturing idiosyncrasies or noise that are not indicative of the overall process behavior. Consequently, the resulting schedule may not accurately reflect the true processes and could be unsuitable for handling novel situations or changes in factory resources.

Moreover, overfitting can lead to overly optimistic schedules since the extracted sequences are based on limited available data. When confronted with real-world scenarios beyond the scope of the gathered data, these schedules may not accurately represent the actual process dynamics. To minimize this risk, users have the option to edit the suggested sequences, allowing for manual adjustments and ensuring greater alignment with the specific real-world requirements and constraints.

Another concern involves the potential of having poor quality data, which may result in invalid indications of available resources or inaccurate timestamps for activity execution. As a result, the Petri net returned by the PM algorithm may be incorrect, resulting in the provision of inaccurate information. For instance, in one of tests carried out in the application, the algorithm suggested activities executed in parallel based on the real data used, even though such parallel execution may not be feasible in real-life scenarios.

This problem increases the complexity of data cleansing and standardization or uniformization. It is a challenging task to ensure data quality, consistency, and compatibility across several sources.

# Chapter 7

# Conclusion and Future Work

The primary objective of this work was to establish the feasibility of employing Process Mining as a means to extract real data and gain insights into how the mold industry's manufacturing processes are executed. Our aim was to leverage this knowledge to enhance the scheduling of activities in the mold manufacturing industry using an adapted approach based on Choueiri and Santos [2021]. By integrating Process Mining techniques with a Genetic Algorithm, we wanted to generate more accurate schedules that effectively allocate resources, leading to improved resource usage and overall operational efficiency.

For the development of our approach, we began by gaining an in-depth understanding of how the processes on the production floors and within the overall business operate. This understanding allowed us to standardize and clean the event logs, as well as concatenate them, since event logs are generated when machinery sensors activate during a process execution.

Following this, the data was organized into mold families, which represent molds with similar purposes and functions within the same industry area (e.g., families for coffee machines, automobile pedals, automobile top covers, and others). By filtering the data based on attributes such as the number of cavities, the number of parts, and specific part types, we were able to identify more structured processes. Using the Inductive Miner Infrequent algorithm, we considered the frequency of each activity, detected deviations from expected processes, and understand the various ways molds can be manufactured.

With this knowledge, we developed an algorithm to extract a feasible sequence from the Petri net generated by the Process Mining algorithm. This algorithm was designed to generate a stochastic output, considering that processes on the production floor are non-deterministic. It was also prepared to accommodate the possibility of activities being executed in parallel and featuring more complex conditions. Given the variability in factory processes, we aimed for a neutral and generic approach, making it adaptable for use in other factories within the industry.

Subsequently, the feasible machines and their corresponding durations were assigned to this sequence to determine the optimal schedule for a part of a mold.

These durations were then converted into confidence intervals, enabling the creation of three scheduling scenarios, an optimistic, a realistic, and a pessimistic scenario by using the margins of the confidence intervals, with the average serving as the basis for the realistic scenario.

Finally, the sequence and its aggregated data is translated into a problem tailored for a Genetic Algorithm. This Genetic Algorithm is designed with the capability to schedule multiple sequences concurrently, taking into account unique due dates for each part and considering machine intervals already occupied by ongoing processes. It aimed to determine the best optimal schedule for each scenario while complying to the provided sequences, resource constraints, and durations.

This approach was integrated into the Prom4Prod project, offering factory managers and workers access to a dedicated frontend page. They can select specific parts, along with their respective molds and mold families, to obtain the three scheduling scenarios. This empowers users to evaluate the effectiveness of the scheduling process under various conditions, enabling informed decision-making and production process optimization.

The conducted tests, including validation with real data, a case study comparing a part previously manufactured in the factory, and a performance test assessing the system's ability to schedule multiple parts simultaneously, revealed the potential of generating viable and competitive schedules using the Petri net output of the Process Mining algorithm.

## 7.1 Future work

Our approach shows promising potential to significantly aid decision-making in the mold industry, particularly due to its applicability in manual and *ad hoc* processes. However, further development and exploration are necessary to fully harness its benefits.

One approach for future work involves incorporating additional domain knowledge into the sequence extraction process. Leveraging the business understanding and expertise of workers, we can enrich the input data to the genetic algorithm and fine-tune the sequence extraction approach.

Refining the rules and constraints used in the sequence extraction process with additional domain knowledge will enable us to derive more meaningful sequences from the process mining data. By incorporating worker expertise, we can better capture the intricacies and nuances of actual production processes, resulting in more accurate and realistic sequences for scheduling.

Another promising direction for future research lies in exploring the application of machine learning techniques to enhance the sequence extraction process. Instead of solely relying on probabilities of activity frequencies, machine learning could help capture the common order of activities found in the process mining data. This approach could potentially lead to more realistic and reliable sequences, contributing to improved scheduling outcomes.

Furthermore, the adaptability of our approach offers exciting opportunities to explore alternative metaheuristic algorithms beyond the Genetic Algorithm for obtaining the fittest schedule. Experimentation with other optimization techniques could shed light on new and potentially more efficient ways of finding optimal schedules based on the extracted sequence.

Through these enhancements, we can further strengthen the performance and adaptability of our approach, unlocking even more significant potential for aiding decision-making and improving scheduling outcomes in the mold industry.

As the mold industry continues to evolve, additional efforts can be devoted to test and validate various approaches that leverage process mining and consider unique production constraints and requirements. Continuous research and development in this domain hold the potential to revolutionize scheduling practices, driving productivity, efficiency, and competitiveness in the mold manufacturing sector.

In conclusion, this study highlights the significant value of using Process Mining as a powerful tool to obtain critical information for scheduling processes in the mold industry. The approach removes the more *ad hoc* technique of manually performing schedules for parts to be manufactured, enabling the industry to make informed decisions based on real data-driven insights. Through the adoption of Process Mining and advanced optimization techniques, the mold industry can achieve enhanced efficiency, resource allocation, and overall productivity, ultimately fostering its continued growth and competitiveness in the manufacturing sector.

# References

Hamzah Alaidaros, Mazni Omar, and Rohaida Romli. The state of the art of agile kanban method: challenges and opportunities. *Independent Journal of Management & Production*, 12(8):2535–2550, 2021.

Anthony Brabazon, Michael O'Neill, and Seán McGarraghy. *Natural computing algorithms*. Springer-Verlag Berlin, 2016.

Juan Pablo Caballero-Villalobos, Gonzalo Enrique Mejía-Delgadillo, and Rafael Guillermo García-Cáceres. Scheduling of complex manufacturing systems with petri nets and genetic algorithms: A case on plastic injection moulds. *The International Journal of Advanced Manufacturing Technology*, 69(9-12):2773–2786, 2013. doi: 10.1007/s00170-013-5175-7.

Peter Chapman, Janet Clinton, Randy Kerber, Tom Khabaza, Thomas P. Reinartz, Colin Shearer, and Richard Wirth. *CRISP-DM 1.0: Step-by-step data mining guide*. SPSS, 2000.

Runwei Cheng, Mitsuo Gen, and Yasuhiro Tsujimura. A tutorial survey of job-shop scheduling problems using genetic algorithms—i. representation. *Computers & Industrial Engineering*, 30(4):983–997, 1996. ISSN 0360-8352. doi: https://doi.org/10.1016/0360-8352(96)00047-2. URL `https://www.sciencedirect.com/science/article/pii/0360835296000472`.

Alexandre Checoli Choueiri and Eduardo Alves Portela Santos. Multi-product scheduling through process mining: bridging optimization and machine process intelligence. *Journal of Intelligent Manufacturing*, 32:1649–1667, 8 2021. ISSN 15728145. doi: 10.1007/s10845-021-01767-2.

Angelo Corallo, Mariangela Lazoi, and Fabrizio Striani. Process mining and industrial applications: A systematic literature review. *Knowledge and Process Management*, 27(3):225–233, 2020. doi: 10.1002/kpm.1630.

B. Efron. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1):1 – 26, 1979. doi: 10.1214/aos/1176344552. URL `https://doi.org/10.1214/aos/1176344552`.

A.E. Eiben and J.E. Smith. *Introduction to Evolutionary Computing*. Springer Berlin Heidelberg, 2015. ISBN 978-3-662-44873-1. doi: 10.1007/978-3-662-44874-8.

Institute for Applied Information Technology Fraunhofer Institute. Understanding process mining, Jan 2021. URL `https://pm4py.fit.fraunhofer.de/getting-started-page#loading-data`.

Xiaoyue Fu, Felix T. Chan, Ben Niu, Nick S. Chung, and Ting Qu. A multi-objective pigeon inspired optimization algorithm for fuzzy production scheduling problem considering mould maintenance. *Science China Information Sciences*, 62(7), 2019a. doi: 10.1007/s11432-018-9693-2.

Xiaoyue Fu, Felix T.S. Chan, Ben Niu, Nick S.H. Chung, and Ting Qu. A three-level particle swarm optimization with variable neighbourhood search algorithm for the production scheduling problem with mould maintenance. *Swarm and Evolutionary Computation*, 50:100572, 2019b. ISSN 2210-6502. doi: https://doi.org/10.1016/j.swevo.2019.100572. URL https://www.sciencedirect.com/science/article/pii/S2210650218309015.

Sebastian Gellrich, Marc-André Filz, Johannes Wölper, Christoph Herrmann, and Sebastian Thiede. Data mining applications in manufacturing of lightweight structures. In Klaus Dröder and Thomas Vietor, editors, *Technologies for economical and functional lightweight design*, pages 15–27, Berlin, Heidelberg, 2019. Springer Berlin Heidelberg. ISBN 978-3-662-58206-0.

Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, 1986. doi: 10.1016/0305-0548(86)90048-1.

Marta Castilho Gomes, Ana Paula Barbosa-Póvoa, and Augusto Queiroz Novais. Reactive scheduling in a make-to-order flexible job shop with re-entrant process and assembly: a mathematical programming approach. *International Journal of Production Research*, 51(17):5120–5141, 2013. doi: 10.1080/00207543.2013.793428.

José Fernando Gonçalves and Mauricio G. C. Resende. *Random-Key Genetic Algorithms*, pages 703–715. Springer International Publishing, Cham, 2018. ISBN 978-3-319-07124-4. doi: 10.1007/978-3-319-07124-4_30. URL https://doi.org/10.1007/978-3-319-07124-4_30.

Wen Ren Jong, Han Ting Chen, Yi Hsin Lin, Yu Wei Chen, and Tai Chih Li. The multi-layered job-shop automatic scheduling system of mould manufacturing for industry 3.5. *Computers and Industrial Engineering*, 149, 11 2020. ISSN 03608352. doi: 10.1016/j.cie.2020.106797.

Jung-Ug Kim and Yeong-Dae Kim. Simulated annealing and genetic algorithms for scheduling products with multi-level product structure. *Computers & Operations Research*, 23(9):857–868, 1996. ISSN 0305-0548. doi: https://doi.org/10.1016/0305-0548(95)00079-8. URL https://www.sciencedirect.com/science/article/pii/0305054895000798.

Zineb Lamghari. Process mining: Basic definitions and concepts. *International Journal of Systematic Innovation*, 7:35–45, 2022. ISSN 20778767. doi: 10.6977/IJoSI.202203_7(1).0003.

Seunghoon Lee, Yongju Cho, and Young Hoon Lee. Injection mold production sustainable scheduling using deep reinforcement learning. *Sustainability*, 12:8718, 10 2020. ISSN 2071-1050. doi: 10.3390/su12208718.

Sander J. J. Leemans, Dirk Fahland, and Wil M. P. van der Aalst. Discovering block-structured process models from event logs - a constructive approach. In José-Manuel Colom and Jörg Desel, editors, *Application and Theory of Petri Nets and Concurrency*, pages 311–329, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-38697-8.

Sander J. J. Leemans, Dirk Fahland, and Wil M. P. van der Aalst. Scalable process discovery with guarantees. In Khaled Gaaloul, Rainer Schmidt, Selmin Nurcan, Sérgio Guerreiro, and Qin Ma, editors, *Enterprise, Business-Process and Information Systems Modeling*, pages 85–101, Cham, 2015. Springer International Publishing. ISBN 978-3-319-19237-6.

David J. Lilja. *Measuring Computer Performance: A Practitioner's Guide*. Cambridge University Press, 2000. doi: 10.1017/CBO9780511612398.

Zhifeng Liu, Linghao Liao, Wentong Yang, Jianhua Wang, and Yongsheng Zhao. Genetic algorithm on solving mould enterprise's job-shop scheduling problem. In *Proceedings of the 2010 International Conference on Computing, Control and Industrial Engineering - Volume 01*, CCIE '10, page 38–41, USA, 2010. IEEE Computer Society. ISBN 9780769540269. doi: 10.1109/CCIE.2010.17. URL `https://doi.org/10.1109/CCIE.2010.17`.

Alan S. Manne. On the job-shop scheduling problem. *Operations Research*, 8(2): 219–223, 1960. doi: 10.1287/opre.8.2.219.

D. Mourtzis, M. Doukas, K. Fragou, K. Efthymiou, and V. Matzorou. Knowledge-based estimation of manufacturing lead time for complex engineered-to-order products. *Procedia CIRP*, 17:499–504, 2014. ISSN 2212-8271. doi: https://doi.org/10.1016/j.procir.2014.01.087. URL `https://www.sciencedirect.com/science/article/pii/S2212827114003394`. Variety Management in Manufacturing.

Dimitris Mourtzis and Ekaterini Vlachou. A cloud-based cyber-physical system for adaptive shop-floor scheduling and condition-based maintenance. *Journal of Manufacturing Systems*, 47:179–198, 2018. ISSN 0278-6125. doi: https://doi.org/10.1016/j.jmsy.2018.05.008. URL `https://www.sciencedirect.com/science/article/pii/S0278612518300700`.

Byung Joo Park and Hyung Rim Choi. A genetic algorithm for integration of process planning and scheduling in a job shop. In Abdul Sattar and Byeong-ho Kang, editors, *AI 2006: Advances in Artificial Intelligence*, pages 647–657, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-49788-2.

S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples)†. *Biometrika*, 52(3-4):591–611, 12 1965. ISSN 0006-3444. doi: 10.1093/biomet/52.3-4.591. URL `https://doi.org/10.1093/biomet/52.3-4.591`.

Leonardo Silva. Manual do projectista de moldes, 2015. URL `http://formacao.training.pt/`.

A Tamilarasi and T Anantha. An enhanced genetic algorithm with simulated annealing for job-shop scheduling. *International Journal of Engineering, Science and Technology*, 2(1), 2010. doi: 10.4314/ijest.v2i1.59105.

Agnieszka Trzcionkowska and Edyta Brzychczy. Practical aspects of event logs creation for industrial process modelling. *Multidisciplinary Aspects of Production Engineering*, 1(1):77–83, 2018. doi: doi:10.2478/mape-2018-0011. URL `https://doi.org/10.2478/mape-2018-0011`.

W. van der Aalst, T. Weijters, and L. Maruster. Workflow mining: discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004. doi: 10.1109/TKDE.2004.47.

Wil van der Aalst. *Process Mining - Data Science in Action*. Springer Berlin Heidelberg, 2016. ISBN 978-3-662-49850-7. doi: 10.1007/978-3-662-49851-4.

Wil M.P. van der Aalst. Process mining: discovering and improving spaghetti and lasagna processes. In *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 1–7, 2011. doi: 10.1109/CIDM.2011.6129461.

W.M.P. van der Aalst and C.W. Gunther. Finding structure in unstructured processes: The case for process mining. *Seventh International Conference on Application of Concurrency to System Design (ACSD 2007)*, Jul 2007. doi: 10.1109/acsd.2007.50.

W.M.P. van der Aalst and A.J.M.M. Weijters. Process mining: a research agenda. *Computers in Industry*, 53(3):231–244, 2004. ISSN 0166-3615. doi: https://doi.org/10.1016/j.compind.2003.10.001. URL `https://www.sciencedirect.com/science/article/pii/S0166361503001945`. Process / Workflow Mining.

AJMM Weijters, Wil MP van Der Aalst, and AK Alves De Medeiros. Process mining with the heuristics miner-algorithm. *Technische Universiteit Eindhoven, Tech. Rep. WP*, 166(July 2017):1–34, 2006.

Stephen A White. Introduction to bpmn. *Ibm Cooperation*, 2(0):0, 2004.

Yu Wu, Xin-cun Zhuang, Guo-hui Song, Xiao-dong Xu, and Cong-xin Li. Solving resource-constrained multiple project scheduling problem using timed colored petri nets. *Journal of Shanghai Jiaotong University (Science)*, 14(6):713–719, Dec 2009. doi: 10.1007/s12204-009-0713-z.

Jian Zhang, Guofu Ding, Yisheng Zou, Shengfeng Qin, and Jianlin Fu. Review of job shop scheduling research and its new perspectives under industry 4.0. *Journal of Intelligent Manufacturing*, 30(4):1809–1830, 2017. doi: 10.1007/s10845-017-1350-2.