Bachelor Thesis Industrial Engineering & Management

Optimizing the Scheduling Method for the Metal Plate Production Process

Author
T. Koeleman (Tijn)
s2599279

Supervisors University of Twente
dr. ir. J.M.J. Schutten (Marco)
dr. ir. E.A. Lalla (Eduardo)

Supervisor Voortman
L.S. Stoverink (Luc)

Faculty of Behavioral, Management and Social Sciences (BMS)

August, 2024

PREFACE

Dear reader,

You are about to read my bachelor thesis titled *Optimizing the Scheduling Method for the Metal Plate Production Process*. I conducted the research at Voortman Steel Group in Rijssen on behalf of the DIGI-STEEL department. My period at Voortman provided me with interesting insights in the family-owned company and the steel processing industry at large. Furthermore, conducting this research helped me to learn more about research methodology, machine scheduling and coding in Python. Before proceeding to read my thesis, I would like to express my gratitude to the people who supported and guided me during my research.

First, I would like to thank Luc Stoverink for the great supervision on behalf of Voortman. The meetings we had guided me into the right direction when needed. Also, I would like to thank the team of DIGI-STEEL for the nice working atmosphere at the office in Rijssen. Furthermore, I would like to thank Marco Schutten for being my first university supervisor during the research. The critical look at my work helped a lot to review and improve the research. Next, I want to thank Eduardo Lalla for being my second university supervisor. Lastly, I owe my thanks to my friends and family who supported me during the period of the research.

Enjoy reading my bachelor thesis!

Kind regards,

Tijn Koeleman

MANAGEMENT SUMMARY

We conduct this research at Voortman Steel Group located in Rijssen and some of its clients in the Netherlands. Voortman fabricates machinery for processing steel beams, plates and tubes. This research focusses on the production process of metal plates at Voortman's clients. Currently Voortman's clients determine their production schedule manually and there are little to no policies for dealing with incoming orders. Therefore, Voortman assumes that the performance of their production schedules can be improved in terms of lead times and in-process inventory. With longer lead times, fewer clients can be served in the same timespan if the work in progress remains the same. When the lead times are shortened, the average in-process inventory level decreases overall as well. Therefore, less space is needed in the warehouse and there is space for more machinery or other purposes.

To research how we can make the production process more efficient, we formulate the following research question:

"How can the processing of metal plates be scheduled more efficiently in terms of inventory levels and lead times?"

The first stage of the research consists of interviewing production planners at Voortman's clients and visiting production lines to get a better image of the scheduling process and the machine environment. Voortman has three types of clients. For this research steel distributors are chosen as a target clients. Steel distributors act as suppliers in the steel processing chain often used to outsource production steps. Therefore, steel distributors aim to maintain short lead times while ensuring quality end products. Currently, production schedulers at steel distributors determine their schedule by prioritizing jobs that are due earlier, which resembles the earliest due date dispatching rule (EDD). Their production process consists of five stages with sometimes machines in parallel. These production stages are: cutting, bevel cutting, deburring, drilling/tapping and bending.

Next, we conduct a literature review to gather information for the solution design. Through reviewing literature, we classify the machine scheduling problem we identify at steel distributors. The machine environment of a steel distributor is a flexible flow shop in which some production stages can be skipped. Also, we collect a selection of dispatching rules for which we test the performance in experiments. These dispatching rules are:

- Shortest processing time (SPT)
- Earliest due date (EDD)
- Weighted shortest processing time (WSPT)
- Minimal slack (MS)
- Apparent tardiness cost (ATC)
- Shortest processing time and earliest due date (SPT/EDD)
- Shortest processing time and minimal slack (SPT/MS)

For the weighted shortest processing time, we determine the weight by the remaining time until the due date. Besides that, we tested two combinations of dispatching rules which were SPT/EDD and SPT/MS. For the formula for ATC a value for k must be determined. Therefore, we run experiments in which we tune the value to find the value for k for which the dispatching rule performs the best. We find that $k = 4$ yields the best performance. Furthermore, we use the same method to determine the value for α for the SPT/EDD rule and SPT/MS rule. This is used in the following formula: $\alpha * SPT + (1 - \alpha) * EDD$. For SPT/MS, the formula is the same, but EDD is substituted by MS. We find that $\alpha = 0.3$ performs the best for SPT/EDD and $\alpha = 0.8$ performs the best for SPT/MS.

Next, we measure the performance of the dispatching rules through conducting experiments on a dataset that consists of orders. This dataset is based on estimates made in consultation with Voortman to resemble the list of orders of a steel distributor. Therefore, we are able to adjust the properties of the datasets to run experiments with different scenarios. In the regular scenario, the ATC rule performs the best with an average lead time of 12,26 working hours, on average 870.94 pieces of in-process inventory and a due date performance of 93.8%. Next, we test how the dispatching rules perform if we alter the standard deviation of the processing times. Also, we test the performance in over- and undercapacity and we test the rules when there the same number of orders arrive every day. From all experiments we conclude that the ATC rule performs the best in most cases. However, the SPT/MS rule sometimes outperforms the ATC rule by a small amount with regards to average lead times and average in-process inventory levels. For both dispatching rules, the due date performance is never lower than 90%, even in undercapacity. Still, the ATC rule outperforms the SPT/MS rule in nearly all cases regarding the due date performance. Furthermore, we notice that these two rules, the WSPT rule and the MS rule always outperform the EDD rule, which is currently used to determine the production schedule.

Based on the research and the results from the experiments, we make the following recommendations:

- Study the dispatching rules using real-world data rather than estimated datasets.
- Consider implementing a tool that utilizes dispatching rules like the apparent tardiness cost to automatically determine the production schedule.
- Conduct a comparable study on the production process at the other types of clients.

This research has limitations, which are important to hold into account when using this research in future work. There are no datasets with real-world data available for the experiments. Therefore, we use datasets based on estimates to test the performance of the dispatching rules. Furthermore, we gather information through interviews and company visits. The information gathered might be subject to bias, because during these interviews, participant tend to create a better image of their company compared to reality, as they are doing business with Voortman. Also, we make some assumptions for the solution model. Finally, the research primarily focusses on the metal plate production process at Voortman's clients. The findings of this research might not be directly applicable to other types of machine environments of other types of clients or a machine environment in a different industry.

CONTENTS

1. INTRODUCTION

This chapter serves as an introduction to the research. Section 1.1 introduces the company Voortman Steel Machinery and DIGI-STEEL which are part of the Voortman Steel Group. Second, Section 1.2 provides a detailed description of the metal plate process. Next, Section 1.3 elaborates on the problem context. Finally, Section 1.4 describes the research design.

1.1. Company description

Voortman Steel Machinery

Voortman Steel Machinery, is a mechanization company that fabricates machinery for metal processing. Parts processed by these machines are for example used for big building projects like the Grolsch Veste in Enschede. The machines can be arranged and delivered as an entire machine line, tailored to the needs of the client. Currently, Voortman has about 700 employees and is still growing. These employees are located at Voortman's headquarters in Rijssen or at one of the other offices spread around the world. These offices all operate under the flag of Voortman Steel Group.

DIGI-STEEL

DIGI-STEEL is a start-up, and part of the Voortman Steel Group, that focusses on software development to optimize the metal processing. This is done by introducing fully cloud-based software solutions for steel processing industries. Their aim is to minimize user interaction and to allow transferring data effortlessly from the digital model of metal parts to the machine. In this research, we address the production process of metal plates on behalf of DIGI-STEEL.

1.2. Metal plate processing

As mentioned earlier, Voortman fabricates machinery for steel processing. One type of products that can be processed using these machinery is metal plates. This process starts with a flat metal plate which is transformed into a part of a machine, ship, or any other object.

A typical plate process starts with a metal plate that needs to be cut. With cutting, a metal plate is placed on a flat surface called the machine bed. The machine moves over the plate and by using for example a plasma or oxy-fuel torch the plate is cut. Figure 1 displays an image of an operating cutting machine from Voortman. From a single plate, multiple shapes are cut. Using the machine, only big holes can be made because there is a high chance that the drill of the machine will break off. Also, bevels can be cut using the cutting machine. A bevel is a diagonal cut. This means that the machine cuts into other materials as well if the parts are too close to each other. Therefore, parts need to be further apart from each other. So fewer parts fit on the machine bed at the same time. If the client wants to save material, he can decide to do the beveling step manually at a separate working station.

Figure 1: A Voortman cutting machine

During the cutting phase, imperfections called burrs appear. So after cutting and beveling, these burrs are removed from the edges and the surface during a process called deburring. Next, small holes and screw threads can be made at the drilling and tapping workstation. This is done by hand, as these holes are smaller than the holes that can be cut by the machine. Following this, the metal plate can be bent using a bending machine. Finally, the part can be coated. For beams, this can be done using a machine. However, for processed plates, there are no machines for coating. Therefore, this task is done by hand. After coating, the processed plate is ready for delivery or assembly. Figure 2 below shows the steps in the production process of metal plates as described in this section.

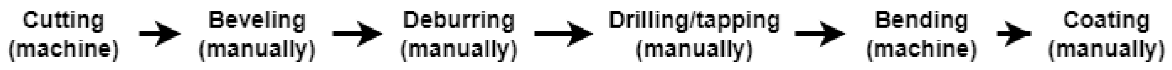

Figure 2: Typical metal plate process

1.3. Problem context

Voortman estimates some clients process 5200 orders on average yearly that consist of 40 pieces on average. Voortman's clients schedule the tasks of the process described above by themselves. However, scheduling tasks requires good scheduling techniques and policies. Yet, Voortman's clients make their production schedule manually. Often, they only use some rule of thumb, like reserving one week for a single task to have enough slack in the production schedule. While we assume that the lead time of an entire order can be shortened to less than a week. Due to a lack of scheduling policies, it is difficult to scale up. For example, when demand increases, there is not enough space in the schedule to allocate more jobs. As a result, lead times are prolonged. When lead times are longer, fewer clients can be served in the same time span if the work in progress remains the same. Also, chances are higher that clients receive their order too late and may switch to competitors that can guarantee a timely delivery. Therefore, less sales and revenue can be generated. Besides that, when inventory levels become higher, more storage capacity is needed, which increases costs. This also means that average handling times between tasks, and thus lead times, become longer as for example some pieces may be difficult to find in a bigger pile of pieces. In the end, these higher costs together with lower revenues lead to suboptimal profits.

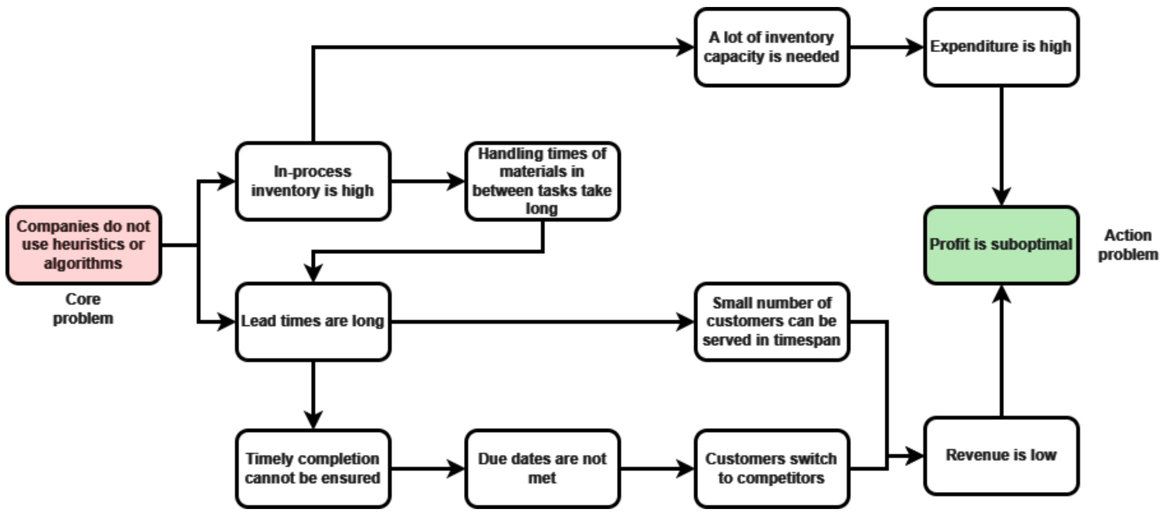
A lot of inventory capacity is needed
Expenditure is high
In-process inventory is high
Handling times of materials in between tasks take long
Companies do not use heuristics or algorithms
Core problem
Profit is suboptimal
Action problem
Lead times are long
Small number of customers can be served in timespan
Timely completion cannot be ensured
Due dates are not met
Customers switch to competitors
Revenue is low

Figure 3: Problem cluster

As depicted in the problem cluster in Figure 3, the problem context points out that manually creating the production schedule is the core problem, as it is the beginning of the problem cluster and there is no other problem causing it (Heerkens & van Winden, 2017). Also, manually creating the production schedule is a problem that can be influenced. By manually creating the production schedule is meant that no scheduling algorithms or heuristics.

1.4. Research design

The objective of this research is to recommend how to schedule tasks in the metal plate process to shorten lead times and decrease inventory. By making the process more efficient, more clients can be served in the same time span and less costs are incurred. Besides that, when less inventory capacity is needed, the space can be utilized for other purposes, such as more machines. In the end, this improvement should result in an increase in profits. The research objective leads to the following formulation of the main research question:

"How can the processing of metal plates be scheduled more efficiently in terms of inventory levels and lead times?"

To enhance manageability of the research, we formulate several sub questions. These sub questions give structure to the thesis as the chapters are divided accordingly.

1. **What is the current situation regarding scheduling of the metal plate process?**
    a. **What does the metal plate process look like?**
    b. **Which methods are currently used to schedule the metal plate process?**

First, we analyze the current situation at Voortman's clients through visiting production lines and conducting interviews with production schedulers and other employees that are closely involved in the process. These employees provide firsthand experience regarding the metal plate process. This involves an explanation of the steps involved in the process, the possible routes through the process and other characteristics of the process. Besides that, production schedulers can show how they schedule tasks in the process. Chapter 2 describes the current situation at Voortman's clients regarding their production processes. Besides that, Chapter 2 contains an explanation of the scheduling method for *steel distributors,* which are one of the three types of clients Voortman serves.

2. **What is discussed in literature regarding scheduling of production processes?**

   a. **What are the key concepts in machine scheduling?**
   b. **Which types of machine environments are distinguished in the literature?**
   c. **Which scheduling heuristics are discussed in literature?**

Second, we conduct a literature review to gain knowledge about machine scheduling. The literature provides a wide range of information on the key concepts of machine scheduling, the different types of machine scheduling problems and methods to address these types of machine scheduling problems. Chapter 3 summarizes the most important findings from the literature review.

3. **How can we improve the scheduling of the metal plate process?**
   a. **How can the current situation be defined as a machine scheduling problem?**
   b. **How can the production process of metal plates be modelled?**
   c. **Which scheduling heuristics are considered suitable for this problem?**

To answer this question, we reassess the current situation using the main findings from the literature review. Using this information, we can specify which type of machine scheduling problem is addressed. From there, the focus shifts to possible solutions in the form of scheduling heuristics. Chapter 4 provides an exploration of these topics.

4. **What is the best scheduling method to use in the metal plate process?**
   a. **Which dispatching rules performs the best?**
   b. **How does the standard deviation of processing times impact the performance of the dispatching rules?**
   c. **How does the number of orders impact the performance of the dispatching rules?**
   d. **How does the arrival pattern of the orders impact the performance of the dispatching rules?**

Finally, we run experiments to test under which dispatching rule performs the best. To do so, we use generated datasets with orders. Because we use generated datasets, we can adjust the parameters to test the performance of the dispatching rules in different scenarios. Chapter 5 presents the setup of the experiments and the results. Given these results, we can draw conclusions regarding the scheduling method of the metal plate production process.

2. CURRENT SITUATION

DIGI-STEEL has little to no information regarding the scheduling methods of their clients. Therefore, we are analyzing the current situation at Voortman's customers. This chapter answers the research question: "What is the current situation regarding scheduling of the metal plate process?" through an analysis at Voortman's clients. First, the different types of clients and their production processes are described in Section 2.1. Second, Section 2.2 discusses how *steel distributors* (a specific type of client) schedule their metal plate process. Finally, the chapter is summarized in Section 2.3.

2.1. Production processes

This section answers the question: "What does the metal plate process look like?" This metal process is different for the different clients Voortman has. Therefore, we explore these three types of processes to select the most suitable process to research more in depth. Voortman divides its clients in three categories: (1) *steel fabrication companies, offshore & energy industry,* (2) *equipment manufacturers* and (3) *steel distributors*. Subsection 2.1.1 discusses the *steel fabrication companies, offshore & energy industry, Next,* Subsection 2.1.2 discusses the *equipment manufacturers* and Subsection 2.1.3 discusses the *steel distributors*. Finally, Subsection 2.1.4 explains which process we use for further analysis.

2.1.1. Steel fabrication companies, offshore & energy industry

Steel fabrication companies make steel structures for large building constructions like office buildings, bridges, production halls, shopping centers and sport halls. Besides that, there is the offshore & energy industry. This industry consists of manufacturing companies that make constructions and components for offshore activities, which are primarily focused on the production of energy. These companies process very large steel parts that are suitable for their constructions.

Process

Both metal plates and metal beams are involved in this process and eventually welded together. The metal plates production process often only consists of a cutting and deburring stage. However for this type of client, both tasks are done simultaneously and considered as a single step. For example, when a metal beam is cut and leaves the cutting machine, the metal beam is deburred immediately. Before the next cut beam leaves the cutting machine, the previous beam already finished the deburring stage. Because of the simplicity of this process, this process can be optimized by *nesting* rather than scheduling. Nesting refers to the process of arranging multiple parts that are cut from a single piece in a way that maximizes the use of material and minimizes the amount of scrap metal. As mentioned earlier, as the last step, the metal plates are welded together with metal beams to become a part of a big structure. Therefore, production schedulers have to make sure that the metal plates and beams are ready at the right time. Also, welding the parts together is an extra step in the process.

2.1.2. Equipment manufacturers

Equipment manufacturers produce equipment for material handling, construction and mining, agriculture, recycling, and many other industries. Examples of equipment include ship cranes, industrial trucks, combine harvesters, shredders, and press containers.

Process

The final products of equipment manufacturers consist of multiple parts that have to be assembled at the end of the process. This means that multiple process flows come together in the final step. When looking at one of these process flows, the process flow resembles the process that is depicted in Figure 2 in Chapter 1. An example of an equipment manufacturer is Company A. This company fabricates recycling installations and sorting systems. Their process consists of the following steps: cutting, deburring, *construction metalwork*, bending, welding and coating. Construction metalwork is a workstation at which multiple steps are done like drilling, tapping and manual beveling. For equipment manufacturers the same holds as for steel fabrication companies, offshore & energy industry, production schedulers have to make sure that all parts of the assembly are ready to be assembled. For equipment manufacturers, this is a bigger challenge because the process consists of more stages prior to the assembly stage. As a result of the complexity of this process, equipment manufacturers have relatively long lead times.

2.1.3. Steel distributors

Steel distributors act as suppliers within the steel processing chain and are mostly used for outsourcing production steps. Other actors in the steel processing chain, like equipment manufacturers, order products from steel distributors for example when they are not able to perform a specific task or do not have enough capacity to produce the desired number of parts on time. Steel distributors sell various products like profiles, plates, tubes and bars.

Process

The production process of steel distributors is comparable to the production process of equipment manufacturers. However, steel distributors do not weld pieces together and do not make assemblies, which means that no flows from other parts join the process. Hence, this makes their production process less complex compared to equipment manufacturers. This allows them to aim for short lead times, which makes them suitable for their clients. For example, when an equipment manufacturer outsources a step in their production process to a steel distributor, it would be unfavorable that this delays their production process. An example of a steel distributor is Company B. They aim to have a lead time of a maximum 1.5 weeks while ensuring consistency in quality.

2.1.4. Focus client for further research

In this research, we aim to improve the production process of metal plates at Voortman's clients. However, the production processes at the three types of clients differ too significantly to conduct a single generic research. Therefore, we have to choose one of the processes from these three types of clients to use for further research. We consider the production process of steel distributors as the most suitable for further research. The first reason for this choice is that steel distributors are focusing more on optimizing their production processes compared to the other clients, especially with regards to lead times and utilization of their machinery. Also, their production process is very similar to the majority of the production process of equipment manufacturers. Therefore, equipment manufacturers can apply the conclusions and recommendations of this research to improve the stages of their process that overlap with the stages of the steel distributors.

2.2. Scheduling methods

This section discusses the methods that are used at steel distributors to schedule the metal plate process. Subsection 2.2.1 elaborates on the aim of the schedule and how this differs from other types of clients. After that, Subsection 2.2.2 discusses how tasks and jobs are prioritized. Next, Subsection 2.2.3, explains how parts are clustered throughout the production process. Finally, Subsection 2.2.4 describes how production schedulers cope with rush orders and rejections of failed parts.

2.2.1. Lead times & time buffers

Overall, steel distributors try to promise short lead times as this makes them more attractive to their clients. These clients often use steel distributors to prevent delays in their production processes. Therefore, lead times are of great importance to them. The other two types of clients focus less on maintaining short lead times, because their production processes are more complex. This difference in aims is noticeable in the way the production process is planned at a steel distributor compared to an equipment manufacturer. For example, Company A (equipment manufacturer) would reserve one week in which a step in the production process can be executed, whereas Company B (steel distributor) tries to keep a maximum time buffer of 1 day between two steps in the production process.

2.2.2. Prioritized scheduling

Besides keeping short time buffers between tasks, Company B often schedules tasks in the production process by due date in ascending order to maintain short lead times. This resembles the *earliest due date* dispatching rule, which is discussed in Chapter 3. However, they do not make use of this dispatching rule as jobs with earlier due dates are not always prioritized. At the beginning of the week, the process scheduler schedules the steps according to his liking, keeping in mind that tasks with an earlier due date have priority.

2.2.3. Clustering

An order that comes in at Company B might consist of sets of different parts. Throughout the process the order does not stay together entirely. Therefore, only the same parts that undergo the same processing steps are kept together. Otherwise, parts are kept idle at a workstation at which they are not processed, whilst waiting for the other parts to be finished. Therefore, parts skip the machines at which no task needs to be executed and are collected together with the other parts after the final step to be shipped to the client.

2.2.4. Rush orders & rejections

Often, production schedulers determine the production schedule at the beginning of the week, based on the received orders and the progress of the other orders. However, it could be the case that a rush order comes in. In a lot of cases, these rush orders have to be finished within a week which impacts the production schedule. Also, it could be the case that something went wrong in the process and a part is not processed correctly. If that is the case, the part will be rejected and the part has to start the process again. If the production scheduler wants to finish these parts on time, he must give these parts priority over other orders to catch up with the wasted time. Therefore, rejected parts and rush orders can be treated the same way. The production scheduler has to postpone tasks with a lower priority in order to fit the rush orders and the parts that were rejected in the production schedule. However, they often do not have a fixed procedure to adjust their schedule.

2.3. Conclusion

In Section 2.1 we presented that he production processes of Voortman's clients vary because of the aim of their business. Steel fabrication companies and the offshore and energy industry only have a single stage in their production process. Because this process can be optimized by nesting rather than scheduling, this type of process is not interesting to explore further in this research.

On the other hand, steel distributors and equipment manufacturers have production processes with a lot of similarities. However, steel distributors are more suitable as a focus client for this research, as they have the most interest in shortening lead times and making their production process more efficient. Also, it is possible to analyze the entire plate production process of steel distributors within the scope of the research as they have a single process flow without interference of other process flows. Besides that, the conclusions and recommendations that result from researching the process of steel distributors can also be applied to the majority of the process of equipment manufacturers.

Section 2.2 discussed the scheduling method of the metal plate process at steel distributors. Company B was used as an example of a steel distributor. In short, Company B tries to promise short lead times by keeping short time buffers between two tasks of the same job, scheduling by ascending due date and not clustering the entire order through the production process. However, they do not make use of a heuristic or dispatching rule that optimizes their production process. Also, they have to adjust their production schedule for rush orders and rejected parts that have to be finished on time, but they lack a procedure to do so.

3. THEORETICAL FRAMEWORK

This chapter covers all the important findings from reviewing literature. The research question that this chapter answers is: "What is discussed in literature regarding scheduling of production processes?"

The first section of this chapter explains the key concepts of machine scheduling. After that, Section 3.2 presents the attributes of tasks. Next, Section 3.3 discusses types of machine environments that are identified in scheduling. Section 3.4 lists some reoccurring restrictions and constraints that impact the nature of a machine scheduling problem. Following, Section 3.5 describes commonly used objective functions in machine scheduling. Finally, Section 3.6 considers some methods to solve machine scheduling problems through heuristics.

3.1. Key concepts of machine scheduling

This section explores the key concepts of machine scheduling. First the two components of a schedule are presented, the jobs and the machines (Subsection 3.1.1). The second subsection (3.1.2) presents how machine scheduling problems are denoted. Next, in Subsection 3.1.3, the difference between online and offline scheduling is explained.

3.1.1. Jobs and machines

The two main components of scheduling problems are jobs and machines. Often machines are also referred to as processor or workstation. The number of jobs and machines are denoted by n and m respectively. These numbers are assumed to be finite in all scheduling problems. Jobs and machines are also used in subscripts for *processing time* (p_{ij}) for example. This variable has the subscript i and j which refer to the machine and the job respectively (Blazewicz, et al., 2001; Leung, 2004; Pinedo, 2022).

3.1.2. Notation

Machine scheduling problems can differ in various ways. The nature of the jobs, the objective function, the type of machines and other restrictions on the schedule influence how a machine scheduling problem must be addressed. Graham, Lawler, Lenstra and Rinnooy (1979), introduce a convenient way to notate different machine scheduling problems. The notation consists of three fields: α (machine environment), β (problem characteristics) and γ (optimality criteria). The first field only contains a single entry, the second field can contain zero to multiple entries and the final field often contains a single entry (Leung, 2004; Pinedo, 2022). These three fields are separated by vertical bars. An example of this notation is $J|prmp|C_{max}$. Sections 3.3, 3.4 and 3.5 discuss the meaning of the fields in this example.

3.1.3. Online and offline scheduling

In machine scheduling, a distinction is made between offline and online scheduling. With offline scheduling, all data such as processing times, release dates and due dates are known beforehand. This allows decisionmakers to determine their schedule at time zero. Sometimes, offline scheduling is referred to as predictive production scheduling or offline-planning (Blazewicz, et al., 2001). On the other hand, there is online scheduling, also known as reactive production scheduling or online control (Blazewicz, et al., 2001). With online scheduling, not all data is known in advance. It could be that not all jobs are known until the last job is released

(Pinedo, 2022). An advantage of online scheduling is that it allows the decisionmaker to make real-time adjustments to the schedule.

3.2. Task attributes

This section describes the attributes of tasks in machine scheduling. These attributes impact the efficiency of a production process. Therefore, it is important to take these attributes into account when determining a production schedule.

Processing time

Processing time is denoted by p_{ij}, which represents the time for how long job j has to be processed on machine i. Sometimes, the subscript of the machine is left out when the processing time is not dependent on the machine or only processed on one machine (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022). Sometimes the p_j symbol is present in the β field. This means that the processing time for each job is restricted. For example, $p_j = 1$ means that all processing times are 1 time unit (Leung, 2004).

Release date

The release date of job j is denoted by r_j. The release date indicates the date when a job arrives in the system and can start its processing (Leung, 2004; Pinedo, 2022). In some cases the release date is referred to as *arrival time* or *ready time* (Blazewicz et al., 2001). When the release date symbol is appears in the β field, job j cannot start processing before the release date. If this symbol is not shown, jobs can start at any time (Leung, 2004; Pinedo, 2022).

Due date

The due date d_j is the date job j is meant to be completed. After the due date, completion of the job is still allowed. Therefore, the due date is not a constraint and not specified in the β field (Leung, 2004; Pinedo, 2022). Although, the due date is important for making use of due date driven *objective functions* which are discussed in Section 3.5.

Deadline

Sometimes deadlines are used which are denoted by $\bar{d}_j$, which is the deadline of job j. This is easily confused with the due date (d_j). The difference is that for deadlines, it is not allowed to finish job j after that point. The deadline is a constraint that can appear in the β field (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022).

Weight

Sometimes, jobs are given a priority factor. This can be done by giving weight to a job. The weight of a job is denoted by w_j. A weight can also represent the actual cost or benefit of a specific job. This priority factor is used for dispatching rules like *weighted shortest processing time,* which is discussed in Section 3.6 (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022).

Setup times

Sometimes a period of time is needed to prepare a machine for the next task, this is called setup time. The duration of the setup time can depend on the sequence of jobs. These are called sequence dependent setup times, denoted by st_{jk} where j represents the first job and k represents the second job. When j equals zero, st_{0k} indicates the setup time before the very first job of the sequence (Pinedo, 2022).

Sometimes the setup time between two jobs is zero. This could occur because two jobs belong to the same job family. Jobs that are in the same job family have little to no setup times between them if they are executed consecutively. Setup times depending on job families are denoted by st_{gh} where g and h represent two job families (Pinedo, 2022).

3.3. Machine environments

This section discusses the most reoccurring types of machine scheduling environments that can be found in literature and therefore answers the question :"Which types of machine environments can be distinguished in the literature?" These machine environments can be divided into two major groups: single-stage scheduling and multiple-stage scheduling, which are discussed in Subsections 3.3.1 and 3.3.2 respectively. The type of machine environment is denoted in the α field of the notation $(\alpha|\beta|\gamma)$ described in Section 3.1.2.

3.3.1. Single-stage scheduling

Single-stage scheduling involves jobs that consist of a single task. This is a fundamental type of scheduling. This type of scheduling can be applied to machine environments that do not have multiple stages. Also, single-stage scheduling is used for analyzing one stage of a multi-stage scheduling problem.

Single-machine

This is the most simple machine scheduling environment. Single-machine systems consist of only one processor and is denoted by 1 in the machine environment field. This type of problem can be used as a building block for more complex problems. For instance, single-machine scheduling is used for analyzing bottlenecks in a multi-processor environment (Blazewicz et al., 2001) Besides that, entire production lines can be analyzed as a single machine, because single-machine scheduling is mathematically more tractable. Therefore, single-machine problems are useful for more general scheduling problems. For example, it can be used for economic lot sizing and capacitated lot sizing (Boctor, 2022).

Parallel machine scheduling

Parallel machine scheduling problems are single-stage scheduling problems that contain a set of multiple machines. For this reason, the scheduler has to choose which task will be performed on what machine. However, the processing times may vary depending on what type of parallel machine scheduling problem is dealt with. This subsection covers three different types of parallel machine scheduling problems: identical machines, uniform machines and unrelated machines.

1. Identical machines

In an identical machine system (P), the processors have the same resource capacity limit (Ji, Hu, Zhang, Cheng, & Jiang, 2022). Therefore, the task processing speeds are equal as well (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022). This means that the processing time of a job is not dependent on which machine is chosen.

2. Uniform machines

For uniform machines (Q) holds that processors differ in task processing speed (Dosa, Fuegenschuh, Tan, Tuza, & Wesek, 2019). However, the processing speeds remain constant and do not depend on the task processed (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022). This is a more difficult problem compared to a parallel machine system with identical machines, as the difference in handling time increases as tasks become bigger (Dosa, Fuegenschuh, Tan, Tuza, & Wesek, 2019).

3. Unrelated machines

For unrelated machines (R), processing times can differ for each machine and processing rates are not fixed within a range (Li, Cote, Coelho, & Wu, 2022). The processing speed depends on the particular task that is processed and the machine (Blazewicz et al., 2001; Pinedo, 2022). Methods that can be applied to parallel machine scheduling for unrelated machines can be applied to identical and uniform machines, but not the other way around. Companies have various reasons to select a machine in a parallel machine system; one machine might be more energy-consuming but more efficient for instance (Wang & Che, 2022).

3.3.2. Multi-stage scheduling

In contrast to single-stage scheduling environments, jobs in multi-stage scheduling environments involve multiple tasks that need to be completed. This means that not only the order of the jobs at the machines has to be determined, but also the order of tasks can vary per job. The possibilities of different orders and routes through the process for jobs depend on the type of scheduling problem. For multi-staged scheduling environments, often it is assumed that buffers between machines have unlimited capacity and jobs do not directly have to resume processing after completion of a task (Blazewicz et al., 2001).

Flow shop

A flow shop (F) consists of a set of machines. Each job has to be processed on each one of the machines and each jobs follow the same order of machines through the process (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022). The first operation is performed on machine M_1, the second is M_2 and the last operation is executed on machine M_m (Garey, Johnson, & Sethi, 1976). Therefore, the number of tasks of a job is equal to the number of machines m.

Job shop

A job shop (J) also consists of a set of m machines. Jobs follow a predetermined sequence of machines. However, these sequences may differ between jobs. Also, the number of tasks may vary for jobs (Blazewicz et al., 2001; Pinedo, 2022). Where the first job might have to be processed on machine 1 first, then machine 2 and finally machine 3, another job might start on machine 2, followed by machine 3 and finish on machine 1.

Open shop

An open shop scheduling problem (O) is similar to a job shop problem. Every job has to be processed on each one of the machines, which means that the number of tasks within a job is again equal to the number of machines. However, there are no restrictions for the routing of every job (Blazewicz et al., 2001; Pinedo, 2022).

Flexible flow shop & flexible job shop

When one more stages in a flow shop or a job shop environment consists of parallel machines, the machine environment is called a flexible flow shop or job shop. The flexible flow shop is denoted by (FF) and the flexible job shop as (FJ). For a flexible flow shop, the job first has to visit stage 1, then stage 2 and so on. In a flexible job shop, jobs have different predetermined routes and orders of machines through the shop (Pinedo, 2022).

3.4. Restrictions & constraints

This section describes some reoccurring characteristics of jobs in machine scheduling. Job characteristics influence the complexity and nature of a machine scheduling problem. These characteristics are specified in the β field of the notation $(\alpha|\beta|\gamma)$.

Preemption

A scheduler may decide to interrupt (preempt) a task. This is called preemption. Not all schedules allow for preemption. When preemption is allowed, a schedules is preemptive, which is notated as $pmtn$ (Blazewicz et al., 2001) or $prmp$ (Pinedo, 2022) in the β field. Otherwise, the schedules is non-preemptive. In some cases a distinction is made between two types of preemption. When a preempted task can resume from the point where the task was interrupted, this is called *preempt-resume*. In other cases, a task cannot be resumed from that point, so it has to start from the beginning again. This case is called *preempt-repeat* (Lambrechts, Demeulemeester, & Herroelen, 2010).

Precedence constraints

For a single machine and parallel machine environments, precedence constraints may appear. In the most general case, this is denoted by $prec$ in the β field (Leung, 2004; Pinedo, 2022). The precedence is pictured in a diagram where each node represents a job and an arrow from job 1 to job 2 indicates that job 1 must be finished before job 2 can start processing (Blazewicz et al., 2001; Leung, 2004). Furthermore, there are some special forms of precedence constraints. When a job has at most one predecessor and at most one successor constraint is denoted by $chains$ in the β field. If each job has at most one predecessor, the constraint is denoted by $outtree$. When the job has at most one successor the constraint is denoted by $intree$ (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022).

No-wait

The no-wait requirement may occur in flow shops and is denoted by nwt in the β field. When the no-wait requirement applies, jobs are not allowed to wait between two successive machines (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022). Therefore, a job often has to wait before starting the first task to ensure that the job can go through the flow shop without waiting between machines.

Other restrictions and constraints

There are many more restrictions and constraints that can be specified in the β field. For a broader overview of more restrictions and constraints, we refer the reader to Pinedo (2022) and Leung (2004).

3.5. Objective functions

The performance of a schedule can be measured in multiple manners depending on the aim of the schedule. Some companies aim to finish jobs as soon as possible for from the moment the job starts the process, whereas other companies aim to have the least number of jobs finished after the due date. These aims can be quantified into objective functions which are specified in the γ field of the notation $(\alpha|\beta|\gamma)$. Objective functions are functions that need to be minimized or sometimes maximized, where an objective to be minimized is always a function of the completion times of the jobs (Pinedo, 2022). This section discusses some frequently reoccurring objective functions, which are split up in two categories: completion-time based objectives and due date-driven objectives.

3.5.1. Completion time-based objectives

This subsection elaborates on objective functions that are based on the completion time of a job. The completion time measures the total amount of time it takes for a task to be completed from the beginning of the schedule. The completion time of job j is denoted by C_j. In most cases, minimizing the completion time decreases costs like holding costs and inventory costs.

Makespan

The makespan, denoted by C_{max}, is the time the last job is finished. The makespan is defined as $\max(C_1, \ldots, C_n)$ where C_j is the completion time of job n (Leung, 2004; Pinedo, 2022). A commonly used objective function is to minimize the makespan which is denoted by just C_{max}. When the makespan is minimized, overall this implies that machines are utilized well (Pinedo, 2022).

Total weighted completion time

The total weighted completion time is denoted by $\sum w_j C_j$ for job j (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022). The sum of the weighted completion times can give an indication of the holding or inventory costs, when the weight is valued by the costs (Pinedo, 2022). Therefore, by minimizing the weighted total completion time, often holding or inventory costs are minimized as well.

3.5.2. Due date-driven objectives

This subsection discusses several objective functions that are due date-driven. Reaching due dates are important for customer satisfaction and delays often incur costs. Therefore, due date-driven objectives focus on minimizing delays and ensuring timely completion of jobs.

Maximum lateness

The lateness of a job is defined as $L_j = C_j - d_j$ for job j. The lateness is positive when the job is completed late and negative when the job is completed early. The maximum lateness is defined as $L_{max} = \max(L_1, \ldots, L_n)$ (Blazewicz et al., 2001; Leung, 2004; Pinedo, 2022). This measures the worst violation of the due dates.

Total tardiness

The tardiness of job j is defined as $T_j = \max(L_j, 0)$ where L_j is the lateness . This means that the tardiness equals zero when a job is finished early or on time and the tardiness is positive when a job is finished late. The total weighted tardiness is denoted by $\sum T_j$ (Leung, 2004; Pinedo, 2022). In some cases, the total tardiness of a job is multiplied by its weight. This can be used to calculate the total weighted tardiness ($\sum w_j T_j$). Just like the total weighted completion time, the total weighted tardiness can give a cost indication by valuing the weight by costs (Pinedo, 2022).

Number of tardy jobs

The total number of tardy jobs is denoted by $\sum U_j$, where $U_j = 1$ when job j is tardy ($L_j > 0$). The number of tardy jobs can reflect the due date performance, which is vital for some companies. Sometimes the weighted number of tardy jobs is calculated. This is denoted by $\sum w_j U_j$ (Leung, 2004; Pinedo, 2022).

3.5.3. Composite objective functions

When there are multiple objectives, a composite objective function can be used. A composite objective function is a combination of two or more objective functions. An example of an composite objective function is $\theta_1 \sum w_j T_j + \theta_2 C_{max}$. This objective function is a combination of the total weighted tardiness ($\sum w_j T_j$) and the makespan (C_{max}). θ_1 and θ_2 are the weights of the two objectives (Pinedo, 2022).

3.6. Heuristics

In deterministic scheduling, one tries to find an optimal schedule, which depends on the chosen objective function. However, most scheduling problems, like the job shop scheduling problem, are considered to not have quick and easy approaches to find an optimal solution are called "NP-hard problems". Approaches that can be used to solve small instances of these problems are not suitable for solving NP-hard problems such as bigger instances of scheduling problems. To come up with an approximate solution, heuristics can be used. Subsections 3.6.1 and 3.6.2 discuss two classes of heuristics, which are *dispatching rules* and *local search algorithms* respectively.

3.6.1. Dispatching rules

One type of heuristics are dispatching rules. A dispatching rule, also known as a priority rule, "is a function of attributes of the jobs and/or the machines" (Pinedo, 2022). Some examples of attributes are listed in Section 3.2. In general, dispatching rules are easy to implement and useful to find a reasonably good schedule.

Local and global dispatching rules

Dispatching rules can be divided into two categories: local and global rules. First, local dispatching rules only make use attributes of the job, the task or the machine. An example of a local dispatching rule is *earliest due date* rule which prioritizes jobs with the earliest due date. On the other hand, global dispatching rules also use attributes of other elements in the process, such as the queue length at the next machine or the processing time of the job on the next machine (Pinedo, 2022).

Static and dynamic dispatching rules

Another way to classify dispatching rules is the distinction between static and dynamic rules. Static rules are not dependent on time. An example of a static dispatching rule is shortest processing time (SPT). This rule only makes use of the processing time, which does not change over time. In contrast, dynamic dispatching rules are dependent on time (Pinedo, 2022). Job characteristics like slack ($slack = d_j - p_j - time$) change as time increments. Therefore, priority values of operations may change during the execution of the dispatching rule.

Composite dispatching rules

When two or more 'elementary' dispatching rules are combined into one rule, it becomes a composite dispatching rule. Static and dynamic dispatching rules can be combined. An example of a composite rule that combines a static and dynamic rule is the Apparent Tardiness Cost (ATC) heuristic. This rule combines the Weighted Shortes Processing Time rule (WSPT) and the Minimal Slack rule (MS) (Pinedo, 2022).

Examples of dispatching rules

Table 1 presents some commonly used dispatching rules, their abbreviations and the attributes used to determine the priority of the job:

Dispatching rule	**Abbreviation**	**Priority**
Earliest Due Date[1]	EDD	Job of which the job has the earliest due date (d_j)
Shortest Processing Time[1]	SPT	Job with the shortest processing time (p_j)
Longest Processing Time[1]	LPT	Job with the longest processing time (p_j)

Weighted Shortest Processing Time[1]	WSPT	Job with the highest weight divided by the processing time (w_j/p_j)
Minimal Slack[1]	MS	Job of which the job has the minimum slack = $\max(d_j - p_j - t, 0)$
Apparent Tardiness Cost[1]	ATC	Job with the highest priority value $I_j(t) = \frac{w_j}{p_j} * \exp\left(-\frac{\max(d_j - p_j - t, 0)}{k * \bar{p}}\right)$ where k is a scaling parameter and $\bar{p}$ is the average processing time of the remaining jobs.
Most operations remaining[2]	MOR	Job with the highest number of operations remaining
Least operations remaining[2]	LOR	Job with the lowest number of operations remaining
Critical ratio[2]	CR	Job with lowest critical ratio $CR = \frac{(d_j - t)}{p_j}$ where p_j denotes the remaining processing time for job j.

Table 1: Dispatching rules [1] (Pinedo, 2022), [2] (Korytkowski et al., 2013)

Generation schemes

For scheduling using dispatching rules, a generation scheme is required to make the schedule. Two schemes can be distinguished: the serial and the parallel generation scheme. In every stage, a generation scheme determines the decision set, which is the set of all schedulable activities. Kolisch (1996), provides an elaborate description of both generation schemes.

1. Serial generation scheme

For the serial generation scheme, the decision set D_s consists of the unscheduled tasks of which the predecessor is already scheduled. The already scheduled tasks are part of the scheduled set S_s. The serial method iterates over the jobs. So the serial method has stages $s = 1, \dots, n$, for which the first stage is the job with the highest priority. The tasks of the job with the highest priority are scheduled at the first point that both the machine and the task are available and there is enough time to process the task. After that, the task is moved to the completed set and the next stage is the job with the highest priority from the decision set (Kolisch, 1996). Figure 4 depicts the flow diagram of the serial generation scheme.

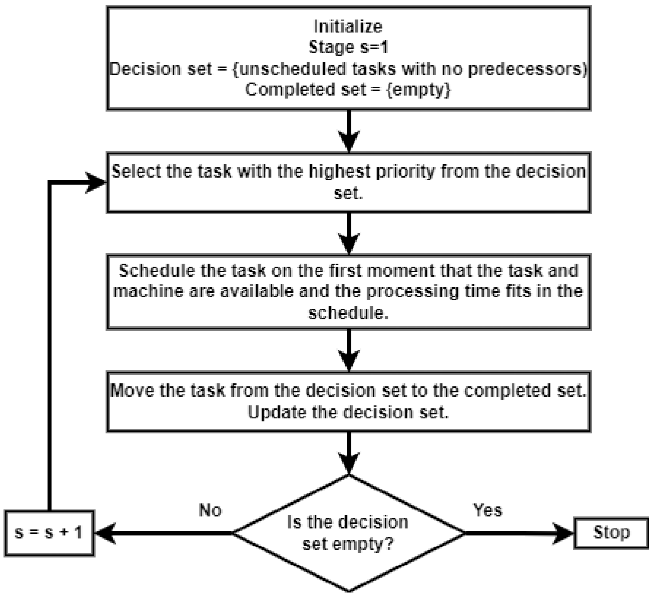
Initialize
Stage s=1
Decision set = {unscheduled tasks with no predecessors}
Completed set = {empty}
Select the task with the highest priority from the decision set.
Schedule the task on the first moment that the task and machine are available and the processing time fits in the schedule.
Move the task from the decision set to the completed set. Update the decision set.
Is the decision set empty?
No
Yes
s = s + 1
Stop

Figure 4: Flow diagram serial generation scheme

2. Parallel method

As described by Kolisch (1996), for the parallel generation scheme, every stage s is associated with a time t_s, where $t_b \geq t_a$ for $b > a$. These times are the times a machine becomes available and are often the completion time of a task. For the tasks that are scheduled, there are two sets: F_s, the set of completed tasks and A_s, the active set of tasks that are scheduled but still active at time t_s. The third set is the decision set D_s which includes all unscheduled tasks that can be scheduled at time t_s. Unscheduled tasks are available, when all preceding tasks are completed and the machine is available at time t_s for the required processing time.

The parallel generation scheme consists of a number of steps. Figure 5 shows the flow diagram of the steps in the parallel generation method. First, the new schedule time is determined. Then, tasks with a completion time equal to the schedule time are removed from the active set A_n. Also, new available tasks will be added to the decision set D_n. Next, a task from the decision set is selected according to the dispatching rule and starts at the current schedule time t_n. This activity is moved from the decision set to the active set. This second step will be repeated until the decision set is empty (Kolisch, 1996).

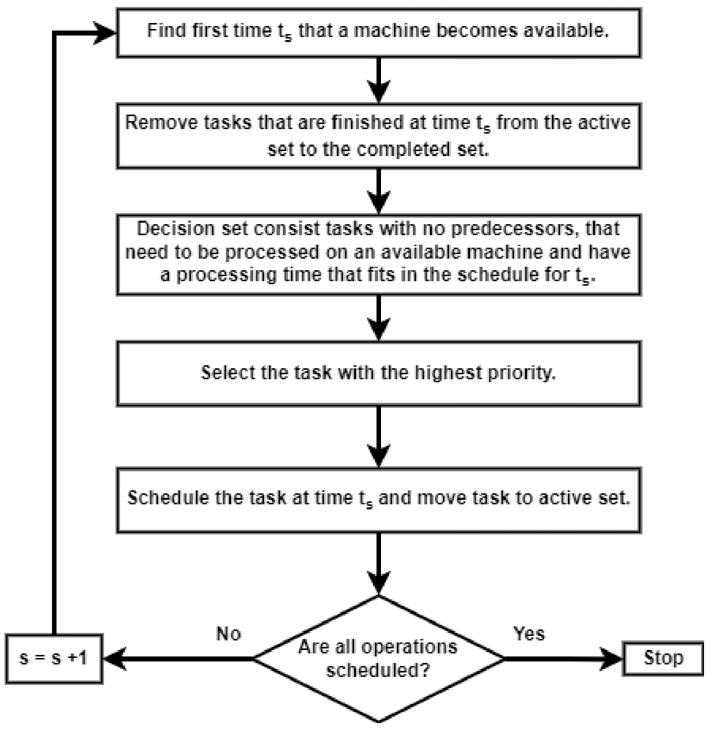
Find first time t_s that a machine becomes available.
Remove tasks that are finished at time t_s from the active set to the completed set.
Decision set consist tasks with no predecessors, that need to be processed on an available machine and have a processing time that fits in the schedule for t_s.
Select the task with the highest priority.
Schedule the task at time t_s and move task to active set.
Are all operations scheduled?
No
Yes
s = s +1
Stop

Figure 5: Flow diagram parallel generation scheme

3.6.2. Local search heuristics

This subsection covers heuristics that are used to find an improved schedule. These heuristics cannot be used for determining a schedule, but they can be used to make improvements. A local search heuristic, often called a meta-heuristic, starts with a feasible schedule. The heuristic tries to iteratively find a better schedule in the *neighborhood* of the current schedule. A schedule is a neighboring schedule, if the schedule can be derived from the current schedule using a specific defined modification. A neighborhood is a collection of these modified schedules (Blazewicz et al., 2001; Pinedo, 2022).

Iterative improvement

A very simple improvement heuristic is the iterative improvement. The heuristic starts with a feasible solution and chooses a better solution in the neighborhood to become the current solution. This process continuous until there is no better solution than the current solution in the neighborhood (Brucker, 2006; Gawiejnowicz, 2008).

Steepest descent

The steepest descent search resembles the iterative improvement algorithm. In steepest descent, all neighbors of the current schedule are considered. From all neighboring schedules, the best schedule is chosen as the schedule. From there, from all neighboring schedules, the best schedule is selected as the schedule until there is no better schedule left (Gawiejnowicz, 2008).

Simulated annealing

In each iteration, simulated annealing uses two schedules, the current schedule and the neighboring schedule. During an iteration, a random schedule is selected from the neighborhood (Brucker, 2006). When the selected schedule is better than the current schedule, the current schedule is changed to the selected schedule. When the selected schedule is worse than the current schedule, there is still a probability that it is accepted as the current schedule (Blazewicz et al., 2001; Pinedo, 2022). This probability is determined by the difference between the values (often the objective functions) of the two schedules and a control parameter. If the neighboring value is higher than the current value, the formula for this probability is as follows:

$$P(move\ to\ neighboring\ schedule) = exp\left(\frac{current\ value - neighboring\ value}{temperature}\right)$$

This control parameter, also called the temperature, decreases while applying the heuristic. When this temperature becomes lower, the chance that a neighbor with a worse value is selected decreases (Pinedo, 2022).

A variant of simulated annealing is the *threshold acceptance method*. This heuristic accepts a neighboring solution if the difference between the current value and the neighboring value is lower than a certain threshold. This threshold is gradually reduced while executing the heuristic (Brucker, 2006).

Tabu-search

The disadvantages of simulated annealing and the threshold acceptance method is that they can come back to solutions that they previously visited. To avoid that, visited solutions can be stored in a *tabu-list* and only solutions that are not on the list are accepted. In a tabu-search, a move is considered from the current schedule to the best schedule in the neighborhood, which is the candidate schedule. However, if the candidate schedule is already visited and is therefore included in the tabu-list, the schedule stays the same. The most recent visited solutions are stored on top of the list. The list often has a maximum number of solutions that can be stored. Therefore, solutions that were visited a certain number of iterations ago are removed from the list and can be accepted again. The tabu search ends when a certain top criterion is met. An example of a stop criterion is a maximum number of iterations (Blazewicz et al., 2001; Brucker, 2006; Pinedo, 2022).

3.7. Summary

In summary, the literature has provided an elaborate theoretical framework that is needed to address the machine scheduling problem at Voortman's clients. First, Section 3.1 examined key concepts of machine scheduling by discussing the main components of machine scheduling problems, the notation and the distinction between online and offline scheduling. Next, Section 3.2 described the which important task attributes need to be taken into account when determining a schedule such as processing times, release dates and due dates. After that, Section 3.3 discussed various machine environments that can either consist of a single production stage or multiple production stages. Following, Section 3.4 discussed restrictions and constraints that might occur in a machine scheduling problem such as preemption, precedence constraints and the no-wait requirement. Subsequently, Section 3.5 discussed the objectives of machine scheduling and how these objectives can be quantified. Most objective functions are based on the completion time of a job or on the due date of a job. However, it is also possible to combine objective functions in a composite objective function. Finally, Section 3.6 explored heuristics that can be used to find an optimal schedule. A schedule can for example determined through using dispatching rules that determine the sequence of jobs according to certain criteria. Besides that, an existing schedule can be improved by a local search heuristic.

We can use the insights gained form this literature review to formulate the solution design in Chapter 4. The exploration of the key concepts, the different machine environments, the restrictions and constraints and the objective functions can be used to classify the machine environment. Furthermore, the discussed heuristics allow us to develop methods to determine a more efficient schedule.

4. SOLUTION DESIGN

This chapter answers the research question: How can we improve the scheduling of the metal plate process?" To answer the research question, we first need to gain a deeper understanding of the of the scheduling problem. Therefore, Section 4.1 discusses the scheduling problem that we address. After that, we formulate our solution model. However, before doing so, Section 4.2 outlines the restrictions and constraints that the solution model must adhere to. Next, Section 4.3 describes the assumptions we make to simplify the scheduling problem into a manageable problem. Then, Section 4.4 describes the dispatching rules that we test. Finally, Section 4.5 introduces the solution model in which we test the dispatching rules form Section 4.4 to find an improved scheduling method for the metal plate production process.

4.1. Scheduling problem

This section discusses the scheduling problem that is identified at steel distributors. First Section 4.1.1 briefly introduces the production process. After that, Section 4.1.2 explains how we classify the machine environment according to literature. Next, Section 4.1.3 examines whether scheduling is done online or offline. Following, Section 4.2 discusses the objectives of this scheduling problem. Then, Section 4.3 discusses how this scheduling problem can be denoted according to the Graham notation that Section 3.1.2 discusses. Finally, Section 4.1.4 provides a lean problem description that summarizes this entire section.

4.1.1. Metal plate process

As described in Chapter 2, the production process of steel distributors consists of six steps. Because the coating stage is outsourced very often, this stage is left out of scope. Figure 6 shows the five remaining stages of the production process. All orders are cut and deburred. The remaining stages are optional depending on the job. The production scheduler determines the production schedule at the beginning of each week. However, there are no policies for determining the scheduling and coping with rush orders. Also, steel distributors often do not track data to analyze the performance of their production process. Voortman assumes that production schedules can be improved by using policies to determine their production schedule.

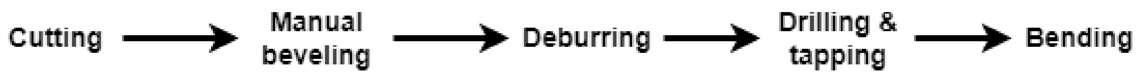

Figure 6: Production process of steel distributors

4.1.2. Machine environment

The machine environment we are addressing consists of a series of 5 machines. Therefore, we consider the machine environment multi-stage. The sequence of tasks of a job is predetermined and tasks are not interchangeable. Therefore, the machine environment is not a job shop or open shop. Because all jobs follow the same order, the machine environment resembles a flow shop. However, for some production stages, there are multiple machines in parallel. Company B has 5 cutting machines for example. Also, jobs do not have to execute every task. For example, one job may be processed on each of the machines whereas another order may not need to be bent and manually drilled. Therefore, we consider this machine environment a flexible flow shop in which tasks can be skipped.

4.1.3. Online/offline scheduling

As Section 2.2 describes, production schedulers of steel distributors determine their production schedule at the beginning of each week. This schedule is based on the orders that have been received until that point in time. Sometimes, during the week, a failed part might be rejected or they might receive a rush order. In this case, the schedule is adjusted real time, but the characteristics of the job such as processing time and due date are known beforehand. Hence, we consider this scheduling problem offline.

4.1.4. Objective

Section 3.5 describes some frequently used objective functions for machine scheduling problems. However, these objective functions only consist of a single aim, such as minimizing the number of tardy jobs. To measure the performance of the improved schedule, we selected multiple key performance indicators. Therefore, defining a single objective function would not suit the aim of this research. To test possible solutions to improve the scheduling of the metal plate production process we consider the results of all key performance indicators. The most important key performance indicators are lead times, in-process inventory levels, due date performance and the utilization of the machines. Voortman aims to have a due date performance around 95% while minimizing the lead times and in-process inventory.

4.1.5. Notation

As Section 3.1.2 describes, machine scheduling problems can be notated the following way: $\alpha|\beta|\gamma$ where α denotes the machine environment, β denotes the constraints and restrictions and γ denotes the objective function. Because we do not specify the objective function, we do not fill anything in in the γ field of the notation. Neither do we fill in the β field of the notation, because there are no general constraints or restrictions to specify in the β field. We classified the machine environment as a flexible flow shop in which tasks can be skipped. Skipping tasks is not conventional for a flexible flow shop we denote the machine environment by FFS^* in the α field.

4.1.6. Summary

In short, steel distributors need to make their production process more efficient in terms of lead times, inventory levels, due date performance and utilization of machines. Their production process consists of five stages: cutting, beveling, deburring, drilling/tapping and bending. The order of these tasks is fixed, however not all jobs have to execute every task. Therefore, we consider the machine environment as a flexible flow shop in which tasks can be skipped. The scheduling is done offline at the beginning of each week. We denote this machine scheduling problem by $FFS^*| - | -$, because we classify the machine environment as a flexible flow shop in which orders can be skipped, there are not constraints for the β field and there is no objective function for the γ field. There is not a single objective function as this scheduling problem has multiple objectives. These objectives are minimizing lead times and in-process inventory, whilst maintaining an acceptable due date performance and machine utilization.

4.2. Restrictions & constraints solution model

To determine an optimal schedule, the schedule must satisfy a set of constraint and restrictions to ensure feasibility. The following points outline the restrictions and constraints that are relevant to the machine scheduling problem we address:

- Preemption is not allowed as it is unconventional for steel distributors to interrupt a task as a result of an adjustment in the schedule.

- There are no precedence constraints between jobs. However, some jobs might have a higher priority than another. Yet, this does not imply that one job is not allowed to start processing.
- It is possible to have in-process inventory between production stages. So, jobs do not have to start the next task after finishing another. Hence, the no-wait requirement does not apply.
- Before a job can start on a machine, time is required to set up the machine.
- When a job is finished on one machine, it has to be transported to the next machine. The time it takes to bring all pieces of a job to the next machine depends on the availability of logistics workers, the number of pieces and the distance to the next machine.
- Every machine can process only one task at a time.
- A job can only be at one machine at a time, because all pieces of a job are clustered together at all times.
- Jobs cannot be scheduled outside the work shifts which are from 8:00 until 16:00.

4.3. Assumptions solution model

We plan to translate the real-world situation of steel distributors into a solution model. However, the real-world situation is too complex to develop a clear framework. To simplify the real-life scenario into a manageable mathematical model that we can analyze, we make the following assumptions:

- Tools, staff and raw materials are always available during work shifts.
- All staff start and end their shift at the same time.
- When a task is interrupted because of a break or the end of the shift, the task can resume without additional setup and processing time.
- Machines of the same type have the same operating speed.
- It takes 30 minutes to move all pieces from one production stage to the next.

4.4. Testing dispatching rules

To determine which scheduling method gives the best performance of the production schedule, we analyze and compare the results of testing several dispatching rules. The dispatching rules we test are chosen in consultation with Voortman. We analyze results for the following dispatching rules:

- Earliest due date (EDD)
- Shortest processing time (SPT)
- Weighted shortest processing time (WSPT)
- Minimal slack (MS)
- Apparent tardiness cost (ATC)
- Composite dispatching rules

Earliest due date

The earliest due date rule (EDD) prioritizes jobs with smaller due dates. Therefore, jobs for which the due date is closer in time will be processed first. This means that the due date performance should improve upon application of this rule. Furthermore, the rule is used to reduce the lateness of jobs as the job that has to be finished the earliest is prioritized. Attaining a high due date performance is one of the objectives of the scheduling problem. Therefore, we consider this dispatching rule suitable for solving this problem.

Shortest processing time

The shortest processing time rule (SPT) sorts the jobs in ascending order of processing time. Mainly, the shortest processing time rule is applied to minimize the average job completion time (Leung, 2004), which is one of the objectives of the scheduling problem.

Weighted shortest processing time (own version)

The shortest processing time (SPT) and the earliest due date (EDD) rules focus on two separate objectives. However, our objective is to both decrease the lead times and improve the due date performance. Hence, we came up with an own version of the weighted shortest processing time rule. The weighted shortest processing time rule (WSPT) prioritizes the job for which the weight divided by the processing time $\left(\frac{w_j}{p_j}\right)$ returns the highest value. To determine the weight of a job we came up with the following formula: $w_j = \frac{1}{\max\,(d_j - t,\ 1)}$. d_j is the due date of job j and t is the current time. The weight of the job is determined according to the amount of time left till the due date of the job. Therefore, this rule can be used to maintain short lead times while accounting for the remaining time until the due date. The denominator cannot be negative, because this would result in a negative outcome of the formula. If we allowed the denominator to be lower than zero, overdue jobs would have the lowest priority.

Minimal slack

The minimal slack rule (MS) prioritizes job with the least amount of slack. Slack is the time that is the time that is left until the due date while accounting for the processing time. The slack of a job is calculated as follows: $slack_j = d_j - t - p_j(remaining)$, where $p_j(remaining)$ is the remaining processing time of job j. Similar to the earliest due date rule, the minimal slack rule prioritizes jobs with an earlier due date. Both rules minimize the risk of jobs completing late. However, the minimal slack rule also accounts for jobs that have longer processing times.

Apparent tardiness cost

The apparent tardiness cost dispatching rule (ATC) is a composite dispatching rule that is often used to reduce costs that are associated with late jobs. The rule combines the conventional weighted shortest processing time (WSPT) rule and the minimum slack (MS) rule into the following formula: $I_j(t) = \frac{w_j}{p_j} * \exp\left(-\frac{\max(d_j - p_j - t, 0)}{k * \bar{p}}\right)$. $\bar{p}$ is the average of the remaining processing times and k is a scaling parameter which we determine through running experiments and analyzing which value of k returns the best performing schedule. Furthermore, all jobs have the same weight, which we set to 1.

Composite dispatching rules

Besides testing the already existing dispatching rules, we introduce two composite dispatching rules. Some dispatching rules, like the shortest processing time (SPT) rule, only take the processing time of a job into account, ignoring factors like the due date. Therefore, we expect that the due date performance is lower when we solely use the SPT rule compared to a combination of the SPT rule and a dispatching rule that takes the due date into account. Therefore, we combine the SPT rule with the EDD and the MS in two composite dispatching rules, which we name the SPT/EDD and SPT/MS rules respectively. The formula for the SPT/EDD rule is $\alpha * SPT + (1 - \alpha) * EDD$. The formula for the SPT/MS rule is very similar to the formula for the SPT/EDD, namely $\alpha * SPT + (1 - \alpha) * MS$. In both formulas α is a weight ranging from 0 to 1. The values for α are determined through tuning the value for α in experiments.

4.5. Solution model

For the solution we use the dispatching rules that Section 4.4 discusses to determine a production schedule and obtain results of the key performance indicators. The solution model consists of two parts. First, Section 4.5.1 discusses how we generate datasets containing lists of order to be scheduled. Second, Section 4.5.2 explains how we use a Python tool to determine production schedules according to the dispatching rules to test their performance.

4.5.1. Generating datasets

We did not succeed in receiving a real dataset or data on processing times from one of Voortman's clients. Therefore, we generate datasets to run experiments with. These datasets contain 52 weeks of orders. The data that we generate per order are: order size, release date, due date and processing time per task. This section discusses how we make estimates in consultation with Voortman on which we base the datasets.

Number of orders & order size

Voortman expects that steel distributors schedule the same hours of tasks as working hours in a week. So if a steel distributor has two cutting machines that are operating the entire week of 40 hours, this means that the workload is approximately 80 hours of cutting tasks per week. This is the expectation for all workstations except for the drilling and tapping workstation, because this is the only workstation that is not busy at all times. Voortman assumes that the workload for the drilling and tapping workstation is 75% of the working hours. Furthermore, the processing times for the cutting stage are estimated by using the averages from data of Voortman's own production hall. The processing times of the remaining stages are deducted from the hours the machine is operating and the estimated orders. Voortman assumes that the mean order size is 40 pieces, with a standard deviation of 10. From the data from Voortman's production hall can be derived that the average processing time at the cutting machine is 0.05 hours per piece. Multiplying the average order size with the average processing time gives an average of 2 cutting hours per order. Based on this estimate, one can also derive the mean number of orders per week as all orders need to be cut. The number of orders per week is 20 per each cutting machine, in a 40 hour work week. The machine environment in which we test the dispatching rules is similar to the machine environment of a steel distributor we visited and interviewed. This machine environment consists of 5 cutting machines. This means that the dataset contains 100 orders (200 hours of cutting workload) per week. Therefore, the datasets contain 5200 orders each.

Due dates & release dates

The due dates in the datasets are randomly distributed over weekdays two weeks from the start of the schedule till two weeks after the end of the schedule. We choose to determine a production schedule for one year starting on July 1 2024, so the earliest possible due date is on July 15 2024. Furthermore, Voortman estimates that:

- 20% of the orders come in 3 weeks prior to the due date. These are orders for which the client expects a lower price, because of the long period before the due date.
- 60% of the orders come in 2 weeks prior to the due date. These are the regular orders of a steel distributor.
- 20% comes in 1 week prior to the due date, which are the rush orders that are processed for a higher price and the orders with failed parts that have to be processed again.

We incorporate this into the dataset by setting the release date to 1, 2 or 3 weeks prior to the due date based on these probabilities.

Processing times & setup times

As mentioned earlier, the processing time at the cutting machine is derived from data that is collected in the production hall. These processing times include the setup times of the machines. Therefore, we use an aggregated setup time per piece that is included in the processing times. Table 2 presents the processing time per stage per piece. In the datasets, the processing times are calculated for the entire order by multiplying the number of pieces by the processing time per piece. For these processing times we use a standard deviation of 10% of the processing time to reduce regularity in the schedule.

Stage	Processing time (h)
Cutting	0.05
Manual beveling	0.067
Deburring	0.02
Drilling & tapping	0.25
Bending	0.083

Table 2: Processing times per piece per stage

Routes through the process

The routes through the process vary per job. Therefore, what the probability is that an order has a task on a certain machine. Every order has to be cut and deburred. Therefore, all orders in the dataset contain processing times for these tasks. To calculate what percentage of orders visits the other three stages (manual beveling, drilling and tapping and bending), we used the same method as for calculating the number of orders. We determined how many orders can be processed on average in the time these machines are available. Next, we calculated the percentage of orders that has to be processed on each type of machine relative to the total number of orders:

- 15% of the jobs are processed on the manual beveling machine.
- 6% of the jobs are processed on the drilling/tapping machine.
- 12% of the jobs are processed on the bending machine.

4.5.2. Python tool

To test the performance of the dispatching rules, we develop a tool in Python that determines the production schedule for 52 working weeks. The production schedule starts July 1, 2024, so it ends on June 27, 2025. Furthermore, the datasets are based on the machine environment of Company B. Therefore, we use the same machine environment which consists of 5 cutting machines, 1 beveling machine, 2 deburring machines, 2 drilling/tapping machines and 1 bending machine. The input for the tool is the dataset that Section 4.5.1 describes. Furthermore, we need to specify the selected dispatching rule and the value for k, in case the ATC rule is selected.

As Section 3.6.1 explains, for implementation of a dispatching rule a generation scheme is required. To solve this scheduling problem we chose to generate the schedule according to the parallel generation scheme. This scheme is suitable for determining a schedule with low idle times of machines, because the parallel generation scheme is designed to schedule a job as soon as the machine is available (Kolisch, 1996). Reaching a high utilization of the machines is one of the goals for improving the schedule. Hence, the parallel generation scheme suits better than the serial generation scheme.

Figure 7 displays how we apply the parallel generation scheme in our Python tool. The tool selects the machine that is available the earliest to schedule a task on that machine. Then, the decision set is created for that machine for that point in time. Jobs that are in the decision set must comply with these criteria:

- The job must be released.
- The job cannot be busy at another machine.
- The job is not in transport between production stages.
- The job has to perform a task on the machine.
- All preceding tasks must be completed.

When the decision set is created, there are two options: there are one or more jobs that can be scheduled or the decision set is empty. When there are one or more jobs to be scheduled, the tool schedules a job according to the selected priority rule. The next time that the machine becomes available is the time when the job is finished on the machine. When the decision set is empty, the tool sets the next point in time to select that machine to the first moment that a job, becomes available for that machine. So either a job that was busy or a new job is released that can start on that machine.

In this scheduling problem, throughout the week, new jobs are received. However, these jobs are only scheduled from the next week, because production schedulers determine the schedule at the beginning of each week. Therefore, when the next point in time a machine becomes available surpasses the end of the week, these new jobs are released to be scheduled. In this research we aim to recommend which scheduling method works the best overall. Therefore, we do not adjust the weekly schedule during the week when an urgent order comes in for example. Also, urgent orders will have a higher priority in most of the dispatching rules that Section 4.4 mentions. Hence, urgent orders will be scheduled sooner overall.

When the next machine and next point in time are set, this loop repeats itself. The loop ends when the earliest time that a machine is available is outside of the given period to schedule tasks. In our case this means that if the time is beyond 16:00 on June 27, 2025, the tool is finished scheduling. The loop does not only end when the next point in time is outside of the timeframe. It also ends when all jobs are scheduled as there are no jobs left to schedule anymore.

The output of the tool is the production schedule, an overview of the unscheduled tasks, an overview of the utilization of the machines and an overview of the other key performance indicators (average lead time, average in-process inventory and due date performance).

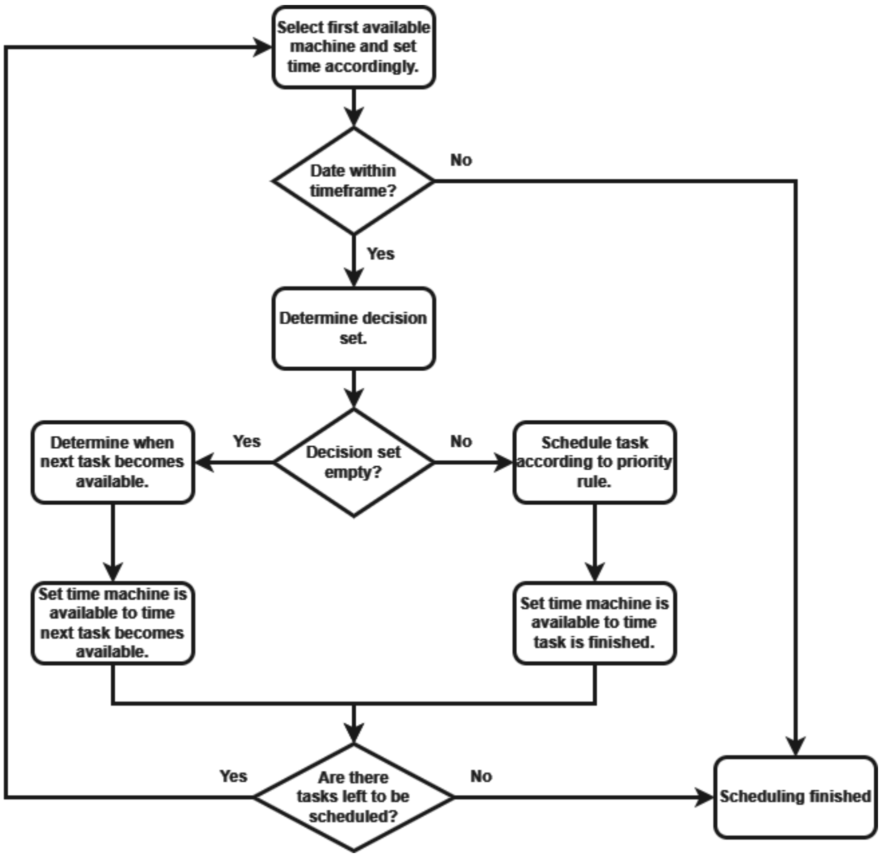
Select first available machine and set time accordingly.
Date within timeframe?
No
Yes
Determine decision set.
Decision set empty?
Yes
No
Determine when next task becomes available.
Schedule task according to priority rule.
Set time machine is available to time next task becomes available.
Set time machine is available to time task is finished.
Are there tasks left to be scheduled?
Yes
No
Scheduling finished

Figure 7: Flowchart scheduling tool

4.6. Summary

This chapter formulated the solution design of the research. First, we described the machine scheduling problem we address. After that, we outlined the restrictions and constraints that the solution model must adhere to. Next, we simplified the scheduling problem by making assumptions to turn the scheduling problem into a manageable solution model. Then, we selected the dispatching rules which we test. These dispatching rules are: shortest processing time (SPT), earliest due date (EDD), our own version of weighted shortest processing time (WSPT), minimal slack (MS) and apparent tardiness cost (ATC). Besides that, we try to combine the SPT with the EDD rule and the SPT with the MS rule in two composite dispatching rules to test whether a combination of these dispatching rules yields an improved performance. Finally, we described our solution model to test the performance of the dispatching rules. First, we generate datasets with a list of orders. Next, we import the dataset into a Python tool that generates a schedule according to the parallel generation scheme. This tool generates the productions schedule and the key performance indicators as output, which we use to compare the performance of the dispatching rules.

5. EXPERIMENTS & RESULTS

This chapter evaluates the solution design through experimenting with the dispatching rules Section 4.4 describes. First, Section 5.1 discusses the experimental design which includes a description of the experiments we run. Next, the following sections present the experiments and the results. Finally, Section 5.8 summarizes the chapter.

5.1. Experimental design

To evaluate the effectiveness of the solution model, we run a set of experiments. In these experiments, we test the dispatching rules that Section 4.4 discusses. This section describes the experiments we run and settings of these experiments. First, Section 5.1.1 provides an overview of the experiments we run. Next, Section 5.1.2 discusses how we compare the results of the experiments through key performance indicators. Finally, Section 5.1.3 describes how we reduce randomness of the outcome and the computational settings of the experiments.

5.1.1. Experiments

To obtain the answer to our main research question, we have to perform a set of experiments. To reduce variability in the results of the experiments, run the tool with 5 datasets. The properties of these datasets may differ per experiment. For the first couple of experiments, we generated 5 datasets with the properties that Section 4.5.1 describes. In the other experiments, properties like the standard deviation of the processing times, the number of orders and the pattern in which orders arrive. The other properties of the dataset always remain the same as Section 4.5.1 describes. Table 3 shows the properties for experiments 1 up to and including 3.

Std. dev. processing time	# orders	Order arrival pattern
10%	5200	Random

Table 3: Properties experiments 1-3

First, before we run experiments to compare the performance of dispatching rules, we determine the value for k that returns the best performance of the ATC rule. We do this by running the tool for different values for k and comparing the results of the key performance indicators. In the next, experiment we use the same datasets to tune the value for α for the SPT/EDD rule. After that, we repeat this with the same datasets for the SPT/MS rule.

When the values for k and α are determined, we start with the first experiment to compare the performance of the dispatching rules. For this experiment, we generate 5 new datasets with the same properties as in Table 3. We view these properties as “regular circumstances”, because Voortman assumes the datasets with these properties resemble the order list of a steel distributor the most. The results of this experiment provide us with a good indication of which dispatching rule performs the best.

Besides testing the dispatching rules under regular circumstances, we also test which dispatching rules perform the best in other scenarios. Therefore, in the fourth experiment, we test the performance of the dispatching rules when the variability of the processing times becomes higher or lower. In the first three experiments, the standard deviation of the processing times was 10%. In these scenarios, we adjust the standard deviation of the processing times to 0%, 5% and 15% and compare the results to the results for the experiment with a standard deviation of 10%. Table 4 displays the properties of the datasets for this experiment.

Std. dev. processing time	# orders	Order arrival pattern
0%, 5%, 15%	5200	Random

Table 4: Properties experiment 4

In the fifth experiment, we test the performance of the dispatching rules in under- and overcapacity. We test these scenarios by adjusting the number of incoming orders by 5%. This means we run experiments for 4.940 and 5.460 orders. As Table 5 shows, the other properties remain the same as in the first three experiments.

Std. dev. processing time	# orders	Order arrival pattern
10%	4.940, 5.460	Random

Table 5: Properties experiment 5

Throughout all experiments, the orders arrived randomly during the weeks. Therefore, more orders can be received in one week compared to another. In this scenario, the same number of orders arrive every day. Therefore, the workload that comes in every week is more equal compared to the previous experiments. Table 6 shows that all properties of the dataset are the same as in the first three experiments except for the order arrival pattern.

Std. dev. processing time	# orders	Order arrival pattern
10%	5200	Constant

Table 6: Properties experiment 6

5.1.2. Key performance indicators

From the schedule that is generated by the tool, we want to analyze the performances of the dispatching rules that we test in the experiments. To analyze and compare the performances of the dispatching rules, we use the following key performance indicators:

- Utilization of the machines; the percentage of time that a machine is busy processing jobs compared to the total time in the schedule. Machines are major investments for steel distributors. Hence, they find it important that their machines are running most of the time.
- Average lead time; the average of the lead times of all finished tasks. The lead time is the time from the moment the first task of a job commences until the final task is finished. Shortening the lead times is one of the objectives of the scheduling problem identified at steel distributors.
- Average in-process inventory; the average amount of inventory that is idle in the process. Steel distributors aim to use space in their warehouses as efficiently as possible. Therefore, they aim to have as little in-process inventory to save space.
- Due date performance: this is the percentage of the jobs that is finished on time. To keep customers satisfied, steel distributors must finish orders on time.

5.1.3. Experimental validation

This section covers the validation of the experiments including randomness of results and computational settings.

Randomness

Datasets generated with the properties that Section 5.1.1 describes might deviate from each other. Therefore, the outcome of the experiment might differ per dataset, while the properties of the datasets are the same. To reduce randomness of the results, we run each experiment with 5 different datasets with the same properties.

Computational settings

The code of the tool is made in Python 3.12.3. Furthermore, we used Microsoft Excel version 2405 to generate and store data. Finally, we run the experiments in the UT-JupyterLab. This is a server hosted by the University of Twente that contains multiple powerful processors, memory cards and graphics cards. For the exact specifications we refer to the website of the University of Twente (University of Twente, sd).

5.2. Experiment 1: determining value for k

First, we determine the value of k for the apparent tardiness cost (ATC) dispatching rule to make sure we analyze the performance of the best performing version of the ATC rule. We do this by running experiments in which we tune the value for k and comparing the results. The first values to run experiments with are integer values. Table 7 presents the key performance indicators for testing integer value.

$k =$	Average lead time (hours)	Average in-process inventory (pieces)	Due date performance (%)
1	14.32	1027.9	95.2
2	13.91	1000.6	95.0
3	13.42	958.4	94.7
4	**13.08**	**934.9**	**94.2**
5	13.32	968.7	93.3
6	13.49	993.7	92.5
7	13.85	1031.3	91.7
10	14.67	1137.4	89.8

Table 7: Results experiments integer values for k

The results show that the due date performance decrease as the value for k increments. Therefore, the probability that the value is greater than 10 will be the best performing value is very low as the due date performance is more than 5% lower than for $k = 1$, while $k = 1$ also performs better regarding the other key performance indicators. Furthermore, the utilization of the drilling and tapping machines also diminishes for higher values of k. The best value for k depends on the preference of the key performance indicators. As some values for k perform better in terms of due date performance, whereas other values perform better in terms of average lead time. Voortman aims to have a due date performance around 95% or higher, therefore we consider the values for k that fulfil this aim. For $k = 4$, the apparent tardiness cost rule performs best in terms of average lead time and average in-process inventory, whilst the due date performance is 94.2%. Therefore, we test values for k close to 4 with one decimal point. We test these values on 5 new datasets because the previous experiment has already shown that the best performing value of k is around 4. Hence, using the same datasets will yield the same results.

The results in Table 8 show that there is little difference in all key performance indicators when the value for k is adjust with 0.1. From this experiment, we notice that $k = 4.0$ again returns the lowest average lead time and the lowest average in-process inventory. Also, the due date performance of 94.2% is acceptable regarding the aim of around 95%. Therefore, for the remainder of the experiments in which we implement the apparent tardiness cost (ATC) rule, the value of k equals 4.

$k =$	Average lead time (hours)	Average in-process inventory (pieces)	Due date performance (%)
3.5	13,26	955,61	94,4%
3.6	13,25	959,86	94,3%

3.7	13,16	946,99	94,3%
3.8	13,21	954,36	94,3%
3.9	13,20	957,13	94,2%
4.0	**13,14**	**949,88**	**94,2%**
4.1	13,17	953,68	94,1%
4.2	13,37	975,55	94,0%
4.3	13,35	978,08	94,0%
4.4	13,25	962,23	93,8%
4.5	13,43	984,12	93,7%

Table 8: Results experiments values with one decimal point for k

5.3. Experiment 2: determining value for α (composite dispatching rules)

As Section 4.4 describes, we also test two composite dispatching rules. The formula for these composite dispatching rules is $\alpha * (Rule\ A) + (1 - \alpha) * (Rule\ B)$. We combine the shortest processing time (SPT) rule with the earliest due date (EDD) and minimal slack (MS) rules. For both rules we determine the value for α that yields the best performance of the dispatching rule. We do this by running experiments and tuning the value for α. We compare the results to analyze which value for α performs the best. First, Section 5.3.1 discusses the results for the SPT/EDD rule. Next, Section 5.3.2 presents the results for the SPT/MS rule.

5.3.1. SPT/EDD

For the SPT/EDD rule we determine the value for α in the following formula $\alpha * SPT + (1 - \alpha) * EDD$. We run experiments with values for α between 0.1 and 0.9. These experiments yield the results shown in Table 9. For higher values of α the due date performance decreases. This can be explained by the fact that when α increases, the SPT rule weighs more and the EDD weighs less. The due date performance is the highest for 0.3 and 0.4. The average lead time and the average in-process inventory are lower for 0.3. Hence, we consider 0.3 the best performing value for α in the SPT/EDD rule and use this value in the remaining experiments.

$\alpha =$	Average lead time (hours)	Average in-process inventory (pieces)	Due date performance (%)
0.9	19,36	1226,73	79,0%
0.8	17,34	1107,31	80,7%
0.7	16,05	1033,78	81,6%
0.6	14,76	945,46	82,1%
0.5	14,23	909,00	82,2%
0.4	13,57	850,94	82,3%
0.3	**13,11**	**820,68**	**82,3%**
0.2	12,78	799,67	82,2%
0.1	12,75	798,34	82,0%

Table 9: Results experiments determining value for α in SPT/EDD rule

5.3.2. SPT/MS

To determine the best performing value for α for the SPT/MS rule, we also run experiments with values between 0.1 and 0.9. Table 10 shows the results for these experiments. We notice that for higher values of α the average lead time and average in-process inventory decreases, because the SPT rule weighs more for higher values of α. For $\alpha = 0.8$, we find an optimum for all three key performance indicators. Therefore, we consider 0.8 the best value to use in the remaining experiments.

$\alpha =$	Average lead time (hours)	Average in-process inventory (pieces)	Due date performance (%)

0.9	13,37	982,61	93,1%
0.8	**13,18**	**934,72**	**94,0%**
0.7	14,12	1005,12	93,6%
0.6	14,84	1076,62	92,8%
0.5	14,96	1097,05	92,6%
0.4	16,11	1188,80	91,6%
0.3	16,73	1179,14	91,3%
0.2	16,86	1162,88	91,6%
0.1	17,00	1164,67	91,8%

Table 10: Results experiments determining value for α in SPT/MS rule

5.4. Experiment 3: Comparing under regular circumstances

Now the best performing values for k and α are determined. We run experiments under regular circumstances for the selected dispatching rules to test which dispatching rule returns the best performing schedule. Table 11 and Figure 20 in Appendix 9.1 display the exact results of the experiment and the utilization of the machines respectively. In the tool, the machines are numbered. For example, a cutting machines are numbered 1.1, 1.2, 1.3, and so on, because it is the first machine in the sequence. Table 12 in Appendix 9.2 serves as a legend for the numbers of the machine. Figure 8 shows that the ATC rule and the SPT/MS rule outperform the other dispatching rules in average lead time. The SPT/MS rule performs slightly better than the ATC rule with a lead time of 12,13 hours compared to 12,26 hours. Also, it can be noticed that the dispatching rules that solely prioritize on one attribute of a job overall perform the worst regarding the average lead time. Figure 9 also shows that the ATC rule and the SPT/MS rule outperform the rest of the dispatching rules with regards to average in-process inventory. The shapes of both graphs are quite similar to each other. Therefore, there seems to be a correlation between the average lead time and the average in-process inventory. When the lead time of jobs are shorter while the processing times remain the same, it means that jobs are idle for shorter periods on average. This means there is also less in-process inventory.

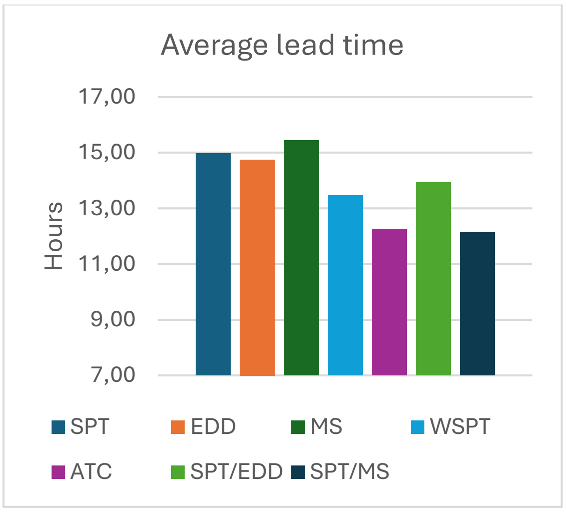

Figure 8: Average lead time under regular circumstances

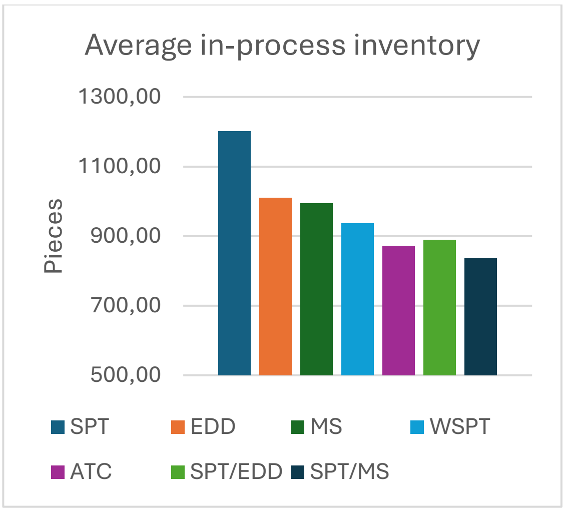

Figure 9: Average in-process inventory under regular circumstances

Figure 10 displays the due date performance of the dispatching rules under regular circumstances. The ATC rule and the SPT/MS rule also outperform the other dispatching rules regarding this key performance indicator. Both rules approximately have the same due date performance, which is 93,8% and 93,7% for the ATC rule and the SPT/MS rule respectively. Therefore, we conclude that the SPT/MS rule outperforms the ATC rule by a small fraction regarding the average lead time and average in-process inventory level. However, this small difference can rely on chance as we ran 5 experiments to determine these results. The similarity between the two rules can be explained by that both rules use the same attributes of a job to determine the priority, namely slack and processing time. Furthermore, we assume that the EDD rule resembles the current scheduling method of steel distributors. We notice that MS and WSPT outperform this dispatching rule as well.

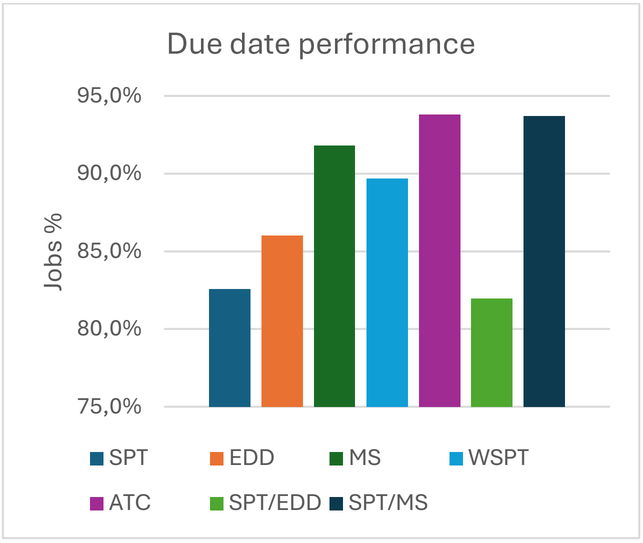

Figure 10: Due date performance under regular circumstances

5.5. Experiment 4: Varying processing times

After running experiments under regular circumstances, we test the performance of the dispatching rules for a scenario in which there is less or more variability in the processing times. This scenario is created by adjusting the standard deviation of the processing times to 0%, 5% and 15%, where 10% is the standard deviation under regular circumstances. Appendix 9.3 presents the exact results and graphs showing the machine utilization. Figure 11 shows the average lead times that result from these experiments. Overall, the average lead time increases when the variability in processing times increases. Also it can be noticed that the performances of the ATC and the SPT/MS rule are very similar and outperforming the other dispatching rules.

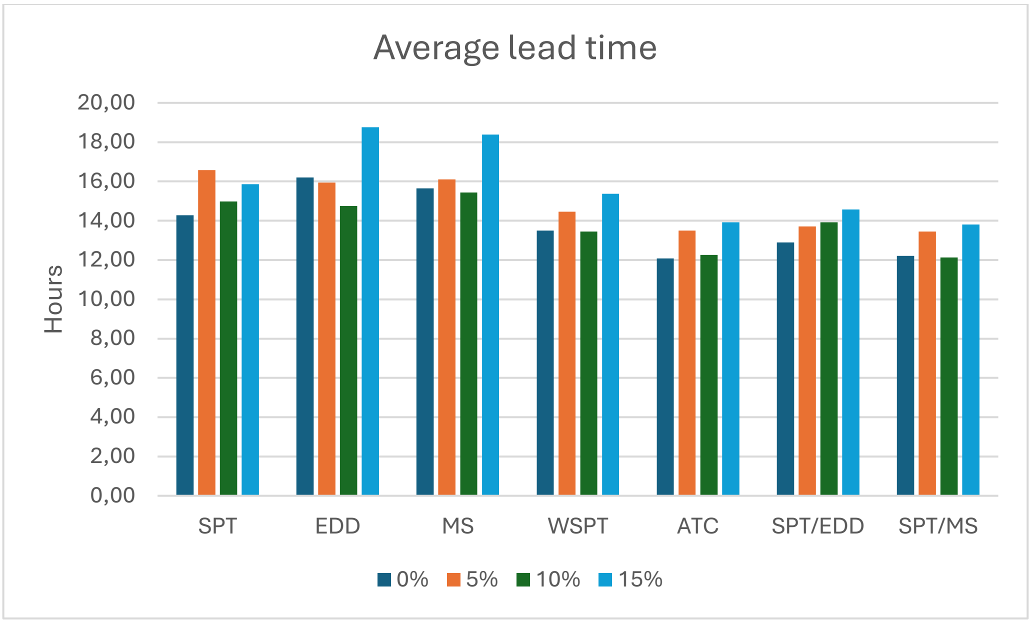

Figure 11: Average lead times for different standard deviations of processing times

Figure 12 depicts the average in-process inventory, which resembles Figure 11 that shows the average lead times. This strengthens the idea that the average lead time and the average in-process inventory are correlated. Surprisingly, the SPT/EDD rule outperforms the ATC and the SPT/MS rule in some cases, while it does never outperform any of both rules regarding the average lead time.

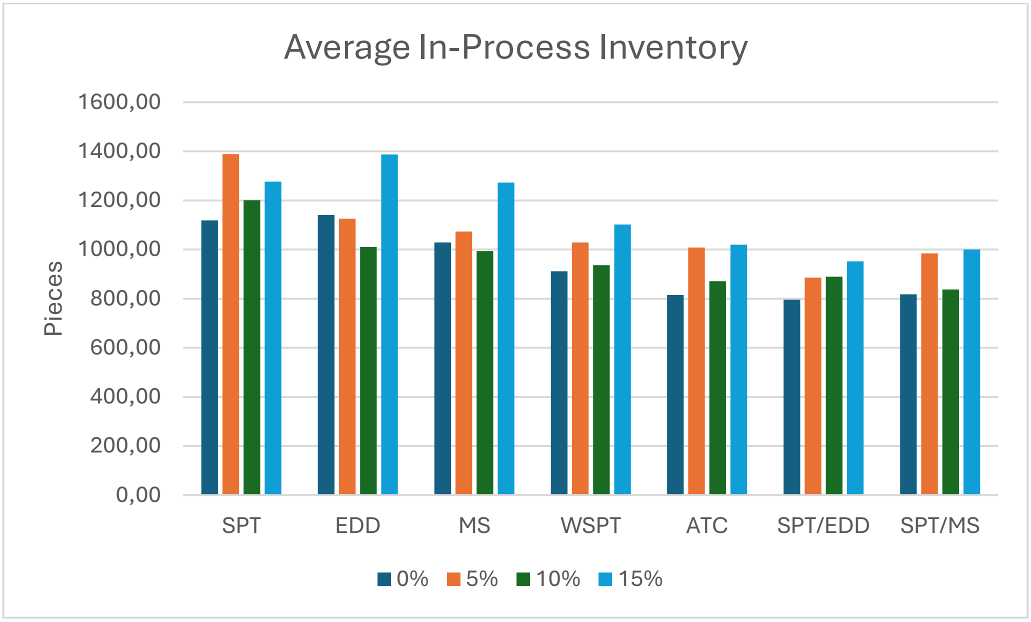

Figure 12: Average in-process inventory for different standard deviations of processing times

Finally, we analyze the due date performance for different levels of variation in the processing times. Figure 13 shows the due date performance of the dispatching rules. From this can be concluded that the SPT/EDD rule performs well regarding the average lead time and the average in-process inventory but performs far worse in terms of due date performance compared to the ATC rule and the SPT/MS rule. It is difficult to judge which of these two rules performs the best for different levels of variability in processing times. For the average lead time and the average in-process inventory, both rules have similar performances and the best performing rule differs per level of variability. However, Table 15 demonstrates that for the due date performance, the ATC rule always outperforms the SPT/MS rule by a small fraction. Therefore, we consider the ATC rule as more suitable over multiple levels of variability in processing times, while there is a very small difference with the SPT/MS rule.

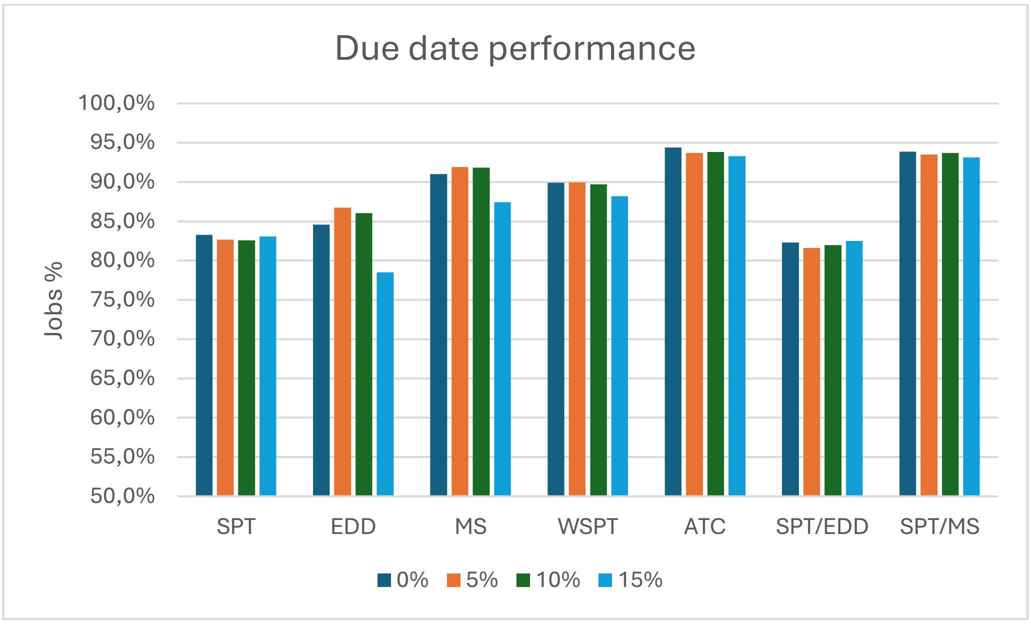

Figure 13: Due date performance for different standard deviations of processing times

5.6. Experiment 5: Over- and undercapacity

Next, we analyze how the dispatching rules perform when 5% more orders come in and 5% less orders come in. We run experiments with 5 datasets of 4.940 orders and 5 datasets of 5.460 orders. The results of these experiments and graphs of the utilization of the machines are shown in Appendix 9.4. Figure 14 shows the average lead times for the different scenarios. We notice that for the EDD and the MS rule the average lead time increases when there are more orders. These two rules differ from the other rules as all other rules also take the shortest processing time into account. Therefore, when a new short order arrives, this order will be processed in a short time which diminishes the average lead time. Furthermore, we notice that the performances of the ATC and the SPT/MS rule are similar again and that those rules are the best performing in both undercapacity and overcapacity. Also, the graph for the average in-process inventory shown in Figure 15 is very similar to the graph of the average lead time

In both graphs, it can be noticed that the SPT rule performs the worst in overcapacity. This can be caused by the fact that the SPT rule now selects jobs with longer processing times more often as all jobs with short processing times are finished. When an order with a shorter processing time comes in, the SPT rule selects this order and ignores the order with the longer processing time.

When 5% more orders are received, it becomes more important to prioritize the right jobs. However, this also means that a lot of jobs cannot be finished as there is more hours in workload than hours in the schedule. From the utilization can be noticed that the utilization of the drilling/tapping machines is very low for the dispatching rules that take the processing time into account. When there are a lot of orders, these rules almost never select the rules with long processing times. Therefore, a lot of these jobs will not be processed at all and be left over at the end of the schedule. Jobs with long processing times often have to be processed at this station. Therefore, the utilization of these machines is very low.

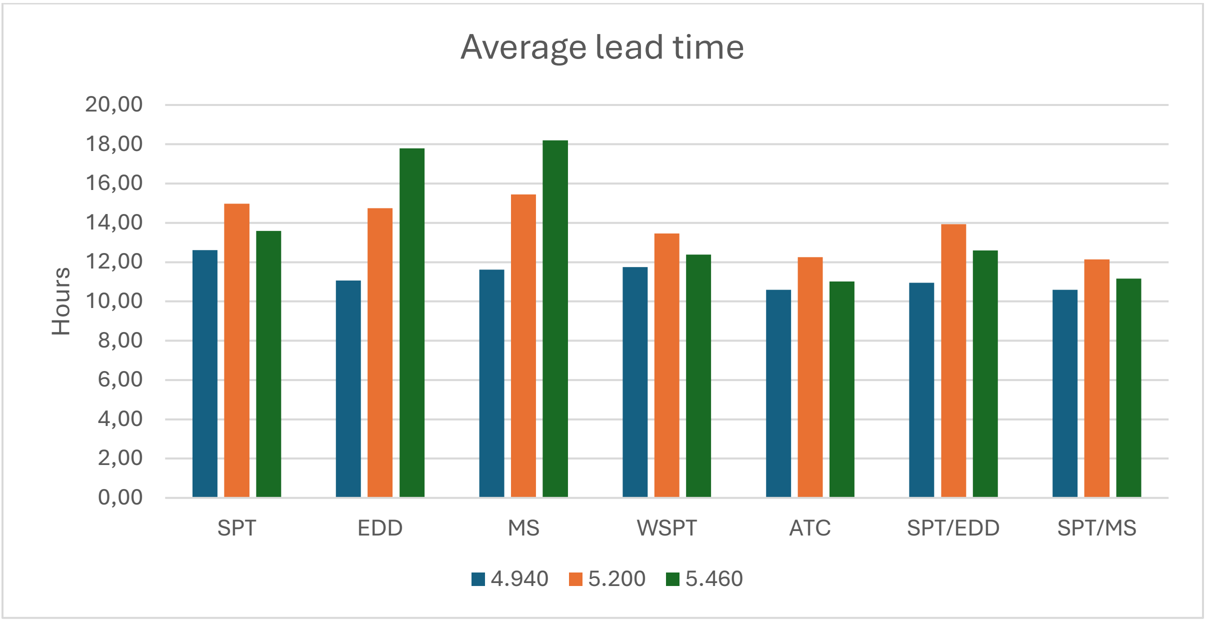

Figure 14: Average lead times for different numbers of orders

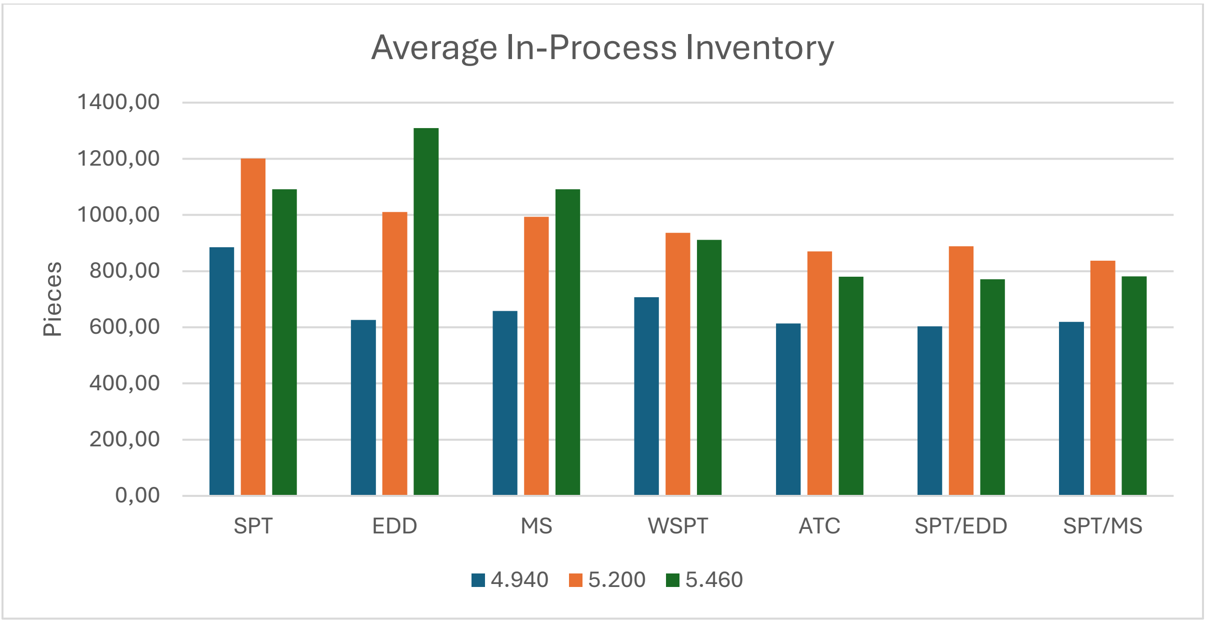

Figure 15: Average in-process inventory levels for different numbers of orders

Figure 16 displays the due date performance in over- and undercapacity. In overcapacity the due date performance is very high overall. Still, the ATC rule and the SPT/MS rule are the two best performing dispatching rules. Overall, the due date performance deteriorates when more orders arrive and thus the workload increases. In undercapacity, we notice that the EDD rule is the worst performing, while this rule only takes the due date of a job into account. The EDD rule selects the job with the earliest due date. However, there are more jobs with the same due date than can be finished before that due date. Therefore, not all jobs are finished on time and the left over jobs still have to be processed. Hence, the backlog of orders increases all the time and the due date performance becomes worse over time.

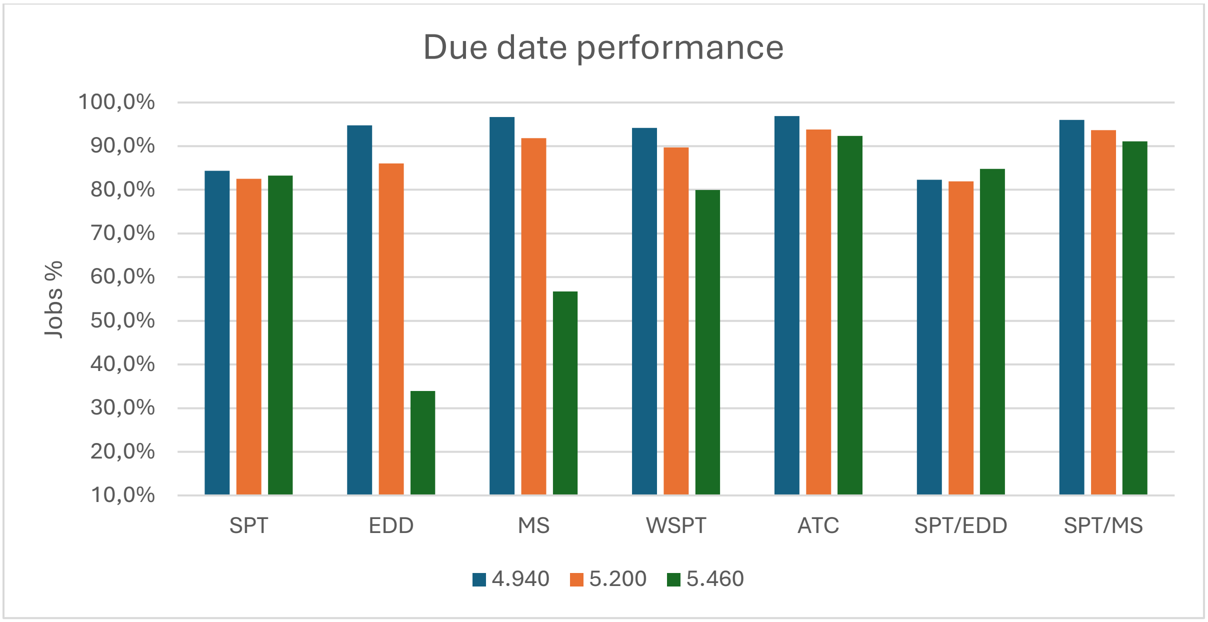

Figure 16: Due date performance for different numbers of orders

From this experiment, we can conclude that the ATC and SPT/MS rule perform the best for the scenarios of over- and undercapacity as well. Table 18 in Appendix 9.4 shows that the due date performance is better for the ATC rule in all cases. Hence, we consider the ATC rule as the most suitable rule in these scenarios.

5.7. Experiment 6: Constant order arrival pattern

In this scenario, we test how the dispatching rules perform when the same amount of orders is received every day. This reduces the variability of workload over the weeks. This means there are less peaks of orders that cause situations of undercapacity. The exact results and graph of the utilization of the machines can be found in Appendix 9.5. Figure 17 and Figure 18 demonstrate that the ATC rule and the SPT/MS rule are the two best performing dispatching rules for a constant order arrival pattern as well. Table 19 shows that the SPT/MS rule outperforms the ATC rule with regards to average lead time and average in-process inventory by a small amount.

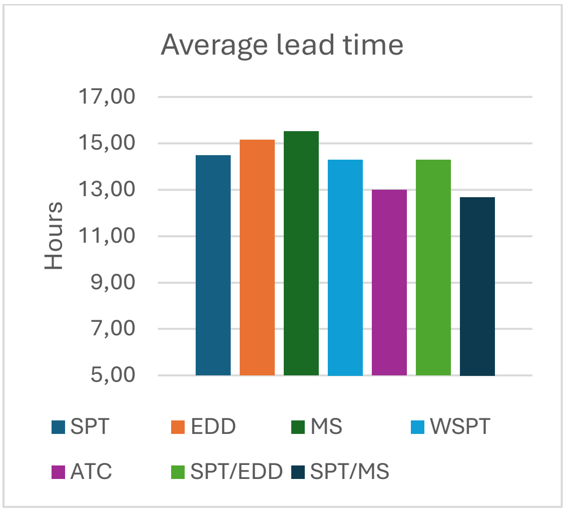

Figure 17: Average lead time constant order arrival pattern

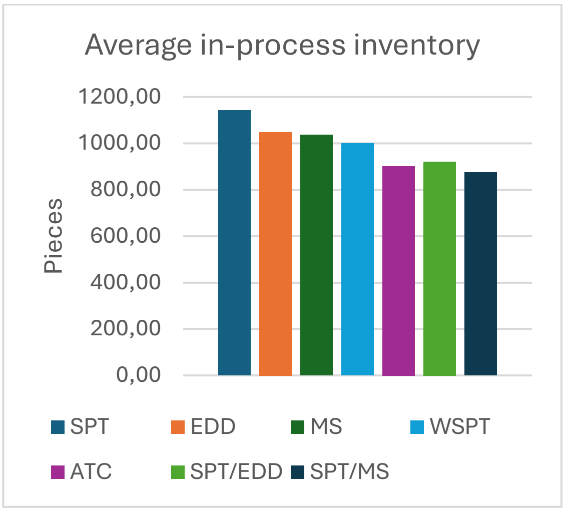

Figure 18: Average in-process inventory constant order arrival pattern

Figure 19 displays the due date performance of the dispatching rules when orders have a constant arrival pattern. From the graph we notice that the ATC rule and the SPT/MS rule again outperform the other dispatching rules. Like for the other experiments, the ATC rule outperforms the SPT/MS rule on a small fraction with regards to the due date performance. In this experiment the ATC rule reached the objective of a 95% due date performance. Therefore, we conclude that the ATC rule is the best performing dispatching rule when there is a constant order arrival pattern.

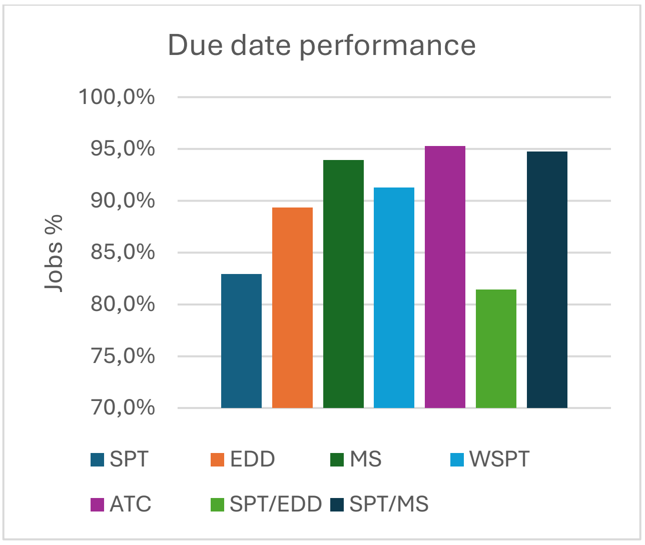

Figure 19: Due date performance constant order arrival pattern

5.8. Summary

We run experiments with the following dispatching rules to test how the method of scheduling at steel distributors can be improved: shortest processing time, earliest due date, weighted shortest processing time (weight based on remaining time until the due date), minimal slack, apparent tardiness cost, SPT/EDD and SPT/MS. First, we determine the value for k for the ATC rule and we determine the values for α for the SPT/EDD and the SPT/MS rule that perform the best. We run the remainder of the experiments with these values. Next, we test these dispatching rules in multiple scenarios. The scenarios we tested are regular circumstances, altering variabilities in processing times, over- and undercapacity and a constant order arrival pattern.

6. CONCLUSION

6.1. Conclusions

This section discusses the main findings and motivates the answer to the main research question:

"How can the processing of metal plates be scheduled more efficiently in terms of inventory levels and lead times?"

We started this research with an analysis of the current situation regarding the scheduling of the metal plate production process at Voortman's clients. Voortman serves three types of clients for metal plates processing: 'steel fabrication companies, offshore & energy industry', 'equipment manufacturers' and 'steel distributors'. We chose to explore the production process of steel distributors and their method of scheduling in more detail. This research involved analyzing of which stages their production process consists; how production schedulers determine the schedule in general; what their aims are when determining the schedule; how they deal with received orders, rush orders and rejected parts and how pieces are clustered throughout the production process.

Next, we conducted a literature review to establish a theoretical framework. The theoretical framework provided relevant theory that we could use to classify the machine scheduling problem. The machine environment is a flexible flow shop in which some production stages can be skipped. Furthermore, the schedule is determined offline at the beginning of each week with on a time horizon of one week. We did not identify a single objective function because the performance of the schedule relies on multiple key performance indicators. Therefore, we measured the performance of the dispatching rules through four key performance indicators: utilization of machines, average lead time, average in-process inventory and due date performance.

To improve the scheduling method for the metal plate production process, we developed a tool which automatically determines the schedule according to a dispatching rule. Before the tool can be implemented, we selected a set of dispatching rules to run experiments with. We experimented with the following already existing dispatching rules: shortest processing time (SPT), earliest due date (EDD), minimal slack (MS), weighted shortest processing time (WSPT) and apparent tardiness cost (ATC). For the apparent tardiness cost, we experimented with multiple values for k to determine which value enhances the performance of the dispatching rule the most.

Besides that, we tested two composite dispatching rules which were SPT/EDD and SPT/MS. The formula for SPT/EDD is as follows: $\alpha * SPT + (1 - \alpha) * EDD$. By running experiments we determined the best performing value for α which is 0.3. SPT/MS has a similar formula: $\alpha * SPT + (1 - \alpha) * MS$. For this dispatching rule we found that $\alpha = 0.8$ yields the best performance.

After completing the set of dispatching rules, we tested the performance of these dispatching rules in multiple scenarios. First, we tested the dispatching rules on a list of orders that was based on estimations in consultation with Voortman. Because this list of orders was based on estimations, we could adjust properties of these datasets to test different scenarios. Therefore, next, we tested how the dispatching rules perform if we altered the standard deviation of the processing times. After that, we tested how the dispatching rules perform in undercapacity and overcapacity. For these cases the number of orders is adjusted by 5% for the same time period. Lastly, we tested in a scenario in which every day the same amount of orders are received. In every experiment the ATC rule and the SPT/MS rule performed the best. The average lead time and the average in-process inventory were nearly the same for all experiments. Also, the due date performances was above 90% for all experiments. However, in most cases the ATC rule outperformed the SPT/MS rule with regards to the due date performance. Therefore, we conclude that the ATC rule performed the best from this set of dispatching rules in most scenarios. Furthermore, we noticed that the WSPT rule and the MS rule outperform the EDD rule very often, while that dispatching rule resembles the current scheduling method the most.

6.2. Recommendations

As a result of the research, several recommendations can be made for Voortman and its clients.

First, Voortman and their clients should consider to study the dispatching rules using real-world data rather than estimated datasets. Such a study will provide a more accurate evaluation of the dispatching rules and their impact on the efficiency of the production process.

Second, Voortman's clients should consider implementing a tool that utilizes dispatching rules like the apparent tardiness cost (ATC) to automatically determine the production schedule. In this way, the majority of the scheduling process is completed in a short time and the production scheduler can analyze the proposed schedule and adjust the schedule if necessary.

Furthermore, we recommend Voortman to conduct a comparable study on the production processes at the other two types of clients. To gain insight in how those processes can be made more efficient in terms of lead times and inventory levels as well. Especially the process for the steel fabrication companies, offshore & energy industry differs a lot from the process we researched.

6.3. Contributions

6.3.1. Theoretical

In this research, we analyzed the performance of the schedule by implementing dispatching rules on a machine environment that is not classified in literature. Therefore, there is little information available in literature on this type of machine environment. Furthermore, we tested two dispatching rules that cannot be found in literature. The first is the weighted shortest processing time for which the weights are based on the remaining time until the due date. The second is a composite dispatching rule that we developed which combines the shortest processing time and the minimal slack rules.

6.3.2. Practical

This research was conducted on behalf of Voortman Steel Group. The solution design in combination with the results from the experiments serve as the practical contribution for the company. Furthermore, Voortman had little to no insights in the scheduling methods at their clients. Therefore, the contextual analysis is a practical contribution to the company as well.

6.4. Limitations

This section covers the limitations of the research. These limitations should be taken into account when implementing the provided solution.

First, there was no dataset with real-world data available for the experiments. Therefore, we used a dataset based on estimates to test the performance of the dispatching rules. These estimates are based on assumptions. For instance, the order sizes are normally distributed with a mean of

40 pieces. In the real world an order may consist of 300 pieces. However, that does not occur in the dataset we tested.

Second, the data on the production processes at Voortman's customers is gathered through interviews and company visits. The information gathered might be subject to bias, because during these interviews, participant tend to create a better image of their company compared to reality, as they are doing business with Voortman. For example, one interviewee claimed that their company maintained a maximum lead time of one week and a half. However, this turned out to be twice as much in some cases.

Furthermore, for the solution model we made several assumptions, such as the constant availability of staff, machinery and raw materials. Also, there were no variations in the time buffers in which a job that has completed a task is transported to the next machine. These assumptions do not hold in a real-world setting. Therefore, this might impact the generalizability of the conclusions.

Finally, the research primarily focusses on the metal plate production process at Voortman's clients. The findings of this research might not be directly applicable to other types of machine environments of other types of clients or a machine environment in a different industry.

7. REFERENCES

Ahmadian, M. M., Khatami, M., Salehipour, A., & Cheng, T. C. (2021). Four decades of research on the open-shop scheduling problem to minimize the makespan. *European Journal of Operational Research, 294*(2), 399-426.

Blazewicz, J., Ecker, K. H., Pesch, E., Schmidt, G., & Weglarz, J. (2001). *Scheduling Computer and Manufacturing Processes.* New York: Springer.

Boctor, F. F. (2022). Single-machine capacitated lot-sizing and scheduling with delivery dates and quantities. *International Journal of Production Research, 60*(24), 7345-7359.

Brucker, P. (2006). *Scheduling Algorithms* (5 ed.). Osnabrück: Springer.

Cooper, D. R., & Schindler, P. S. (2014). *Business Research Method Twelfth Edition.* New York: McGraw-Hill/Irwin.

Dosa, G., Fuegenschuh, A., Tan, Z., Tuza, Z., & Wesek, K. (2019). Tight lower bounds for semi-online scheduling on two uniform machines with known optimum. *Central European Journal of Operations Research, 27*, 1007-1130.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976, May). The Complexity of Flowshop and Jobshop Scheduling. *Mathematics of Operations Research, 1*(2), 117-129.

Gawiejnowicz, S. (2008). *Time-Dependent Scheduling.* Springer.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Rinnooy Kan, A. H. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics, 5*, 287-326.

Ham, M., Lee, Y. H., & Kim, S. H. (2011, June 15). Real-time scheduling of multi-stage flexible job shop floor. *International Journal of Production Research, 49*(12), 3715-3730.

Heerkens, H., & van Winden, A. (2017). *Solving Managerial Problems Systematically.* Enschede, Nieuwegein: Noordhoff Uitgevers.

Ji, M., Hu, S., Zhang, Y., Cheng, T. C., & Jiang, Y. (2022). Parallel-machine scheduling with identical machine resource capacity limits and DeJong's learning effect. *International Journal of Production Research, 60*(9), 2753-2765.

Kelley Jr., J. E. (1963). The critical-path method: Resources planning and scheduling. *Industrial Scheduling*, 347-365.

Kolisch, R. (1996). Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. *European Journal of Operational Research, 90*, 320-333.

Korytkowski, P., Wisniewski, T., & Rymaszewski, S. (2013). An evolutionary simulation-based optimization approach for dispatching scheduling. *Simulation Modelling Practice and Theory, 35*, 69-85.

Lambrechts, O., Demeulemeester, E., & Herroelen, W. (2010). Time slack-based techniques for robust project scheduling subject to resource uncertainty. *Annals of Operations Research, 177,* 186.

Leung, J. Y.-T. (2004). *Handbook of Scheduling: Algorithms, Models and Performance Analysis*. Chapmann & Hall/CRC.

Li, Y., Cote, J.-F., Coelho, L. C., & Wu, P. (2022). Novel efficient formulation and matheuristic for large-sized unrelated parallel machine scheduling with release dates. *International Journal of Production Research, 60*(20), 6104-6123.

Pinedo, M. L. (2022). *Scheduling Theory, Algorithms, and Systems* (6 ed.). Cham, Switzerland: Springer.

Thomé, A. M., Scavarda, L. F., & Scavarda, A. J. (2016). Conducting systematic literature review in operations management. *Production Planning & Control: The Management of Operations, 27(5)*, 408-420.

University of Twente. (n.d.). *Hardware*. Retrieved July 2024, from Jupyter wiki: https://jupyter.wiki.utwente.nl/#hardware

Wang, Y., & Che, A. (2022). Energy-efficient unrelated parallel machine scheduling with general position-based deterioration. *International Journal of Production Research, 61*(17), 5886-5900.

8. GLOSSARY

8.1. Terms

Term	Definition
Apparent Tardiness Costs (ATC)	A dispatching rule which prioritizes jobs according to a formula that combines the weighted shortest processing time (WSPT) and the minimal slack (MS) rules.
Bending	The process of deforming a metal plate to reach a desired angle or curve.
Bevel cutting	The process of cutting diagonally through a metal surface.
Coating	The process of applying a colored layer to the metal surface.
Construction metalwork	A work station where manual drilling, tapping and milling is done.
Cutting	The process of separating or shaping metal plates into a desired shape.
Deburring	The process of removing imperfections (burrs) that appear after cutting from the edges and surface.
Dispatching rule	A heuristic that determines the priority of jobs in a scheduling process.
Earliest Due Date (EDD)	A dispatching rule which prioritizes jobs with earliest due dates.
Flexible Flow Shop (FF)	A flow shop with machines in parallel at one or more stages.
Flexible Job Shop (FJ)	A job shop with machines in parallel at one or more stages.
Flow Shop (F)	A machine environment in which jobs follow a predetermined route for which the order of the machines is fixed.
Job	A collection of tasks that are grouped together. In this research, jobs consist of two or more tasks.
Job Shop (J)	A machine environment in which jobs follow a predetermined route for which the order of machines may vary.
Longest Processing Time (LPT)	A dispatching rule which prioritizes jobs with the longest processing times.
Machine	Equipment used to perform a task.
Machine bed	A flat surface on which a metal plate is placed to be processed.
Makespan (C_{max})	The total time that is required to finish all jobs.
Minimal Slack (MS)	A dispatching rule which prioritizes jobs with the least amount of slack time.
No-wait (nwt)	A constraint where jobs cannot wait between two machines.
Open Shop (O)	A machine environment in which jobs do not follow a predetermined route and the order of machines may vary.
Precedence constraint ($prec$)	A constraint for which certain jobs have to precede others.
Preemption ($prmp\ or\ pmtn$)	The ability to interrupt a task to start another.
Setup Time	The time it takes to prepare a machine for a task.
Shortest Processing Time (SPT)	A dispatching rule which prioritizes jobs with shortest processing times.

Slack	The amount of time that a job can be delayed without completing the job late.
Steel distributors	Steel processing companies that supply steel parts to other companies. Often, companies outsource tasks they are unable to do to steel distributors. For more information, see Section X.
Task	A single unit of work that has to be done. In this research, a task is always part of a job that consists of two or more tasks. Often referred to as operation.
Weighted Shortest Processing Time (WSPT)	A dispatching rule which prioritizes jobs with the highest weights divided by the processing times.
Welding	The process of joining two or more pieces of metal by fusing the material.

8.2. Abbreviations

Abbreviation	Definition
ATC	Apparent Tardiness Cost
EDD	Earliest Due Date
F	Flow Shop
FF	Flexible Flow Shop
FJ	Flexible Job Shop
J	Job Shop
LPT	Longest Processing Time
MS	Minimal Slack
nwt	No-wait
O	Open Shop
$pmtn$	Preemption
$prec$	Precedence constraint
$prmp$	Preemption
WSPT	Weighted Shortest Processing Time

8.3. Variables

Variable	Definition
A_s	$active\ set\ at\ stage\ s$
C_j	$completion\ time\ of\ job\ j$
D_s	$decision\ set\ at\ stage\ s$
d_j	$due\ date\ of\ job\ j$
F_s	$completed\ set\ at\ stage\ s$
i	$machine\ number$
j	$job\ number$
L_j	$lateness\ of\ job\ j$
LT_j	$lead\ time\ of\ job\ j$
M_i	$machine\ i$
m	$number\ of\ machines$
n	$number\ of\ jobs$
p_{ij}	$processing\ time\ of\ job\ j\ on\ machine\ i$
S_s	$scheduled\ set\ at\ stage\ s$
s	$stage\ in\ the\ generation\ scheme$
st_{jk}	$setup\ time\ between\ job\ j\ and\ k$
T_j	$tardiness\ of\ job\ j$
t	$current\ time$
t_s	$time\ at\ stage\ s$

U_j	$1\ if\ job\ j\ is\ tardy, 0\ if\ job\ j\ is\ on\ time$
w_j	$weight\ of\ job\ j$

9. APPENDIX

9.1. Experiment under regular circumstances

Dispatching rule	Average lead time (hours)	Average in-process inventory (pieces)	Due date performance (%)
SPT	14,98	1200,90	82,6%
EDD	14,75	1010,52	86,0%
MS	15,44	993,52	91,8%
WSPT	13,46	936,81	89,7%
ATC	12,26	870,94	93,8%
SPT/EDD	13,93	889,12	82,0%
SPT/MS	12,13	837,60	93,7%

Table 11: Results experiment under regular circumstances

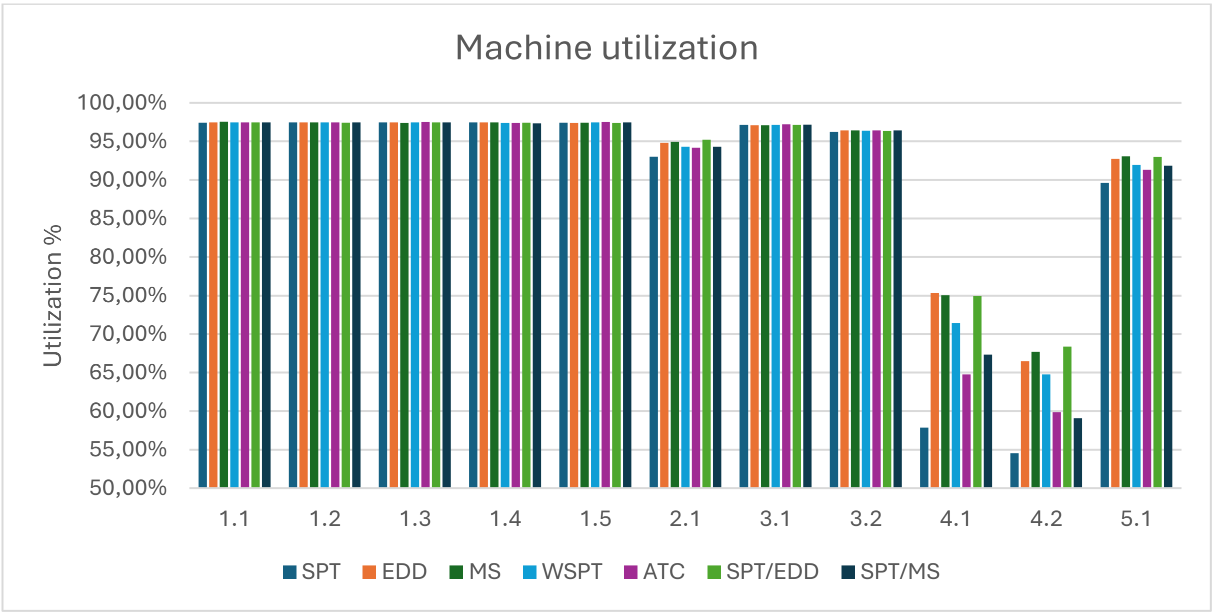

Figure 20: Machine utilization under regular circumstances

9.2. Legend machines

Number	Type of machine
1.1-1.5	Cutting
2.1	Beveling
3.1-3.2	Deburring
4.1-4.2	Drilling/tapping
5.1	Bending

Table 12: Legend machines

9.3. Experiments variability in processing times

	0%	5%	10%	15%
SPT	14,29	16,58	14,98	15,87
EDD	16,21	15,94	14,75	18,77

WSPT	15,65	16,11	15,44	18,39
MS	13,50	14,46	13,46	15,37
ATC	12,08	13,50	12,26	13,92
SPT/EDD	12,90	13,70	13,93	14,57
SPT/MS	12,22	13,45	12,13	13,81

Table 13: Average lead times in hours for different standard deviations of processing times

	0%	5%	10%	15%
SPT	1118,52	1389,14	1200,90	1276,13
EDD	1141,20	1125,58	1010,52	1387,44
WSPT	1029,48	1073,71	993,52	1272,62
MS	911,66	1029,41	936,81	1101,22
ATC	814,84	1008,29	870,94	1020,26
SPT/EDD	796,11	884,88	889,12	952,25
SPT/MS	817,87	985,20	837,60	999,90

Table 14: Average in-process inventory in pieces for different standard deviations of processing times

	0%	5%	10%	15%
SPT	83,3%	82,7%	82,6%	83,1%
EDD	84,6%	86,7%	86,0%	78,5%
WSPT	91,0%	91,9%	91,8%	87,4%
MS	89,9%	89,9%	89,7%	88,2%
ATC	94,4%	93,7%	93,8%	93,3%
SPT/EDD	82,3%	81,6%	82,0%	82,5%
SPT/MS	93,9%	93,5%	93,7%	93,1%

Table 15: Due date performance in % of jobs for different standard deviations of processing times

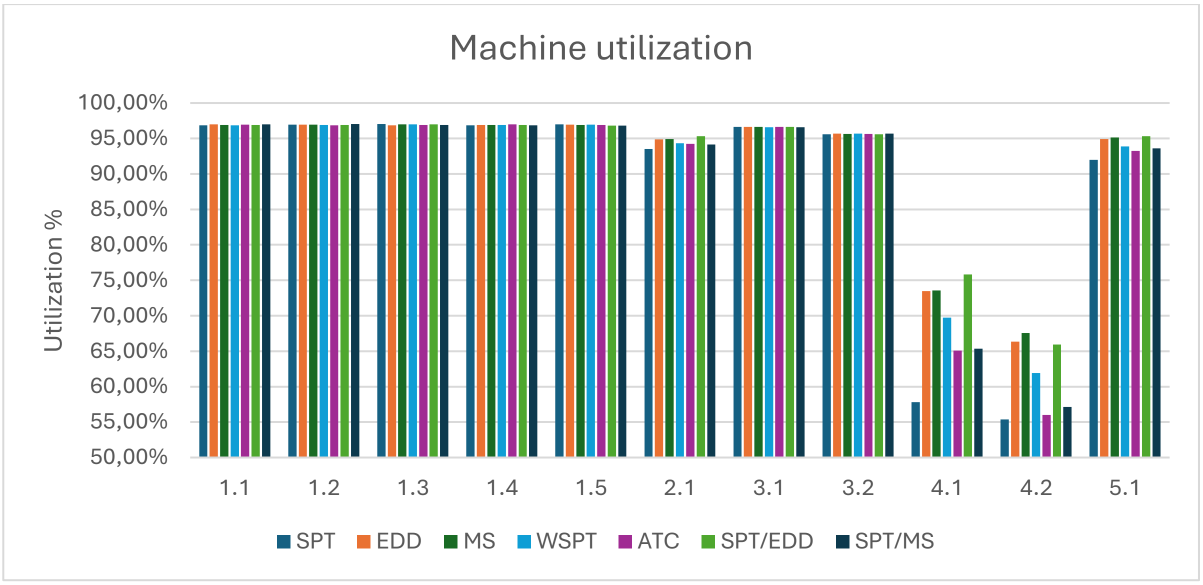

Figure 21: Machine utilization for 0% standard deviation of processing times

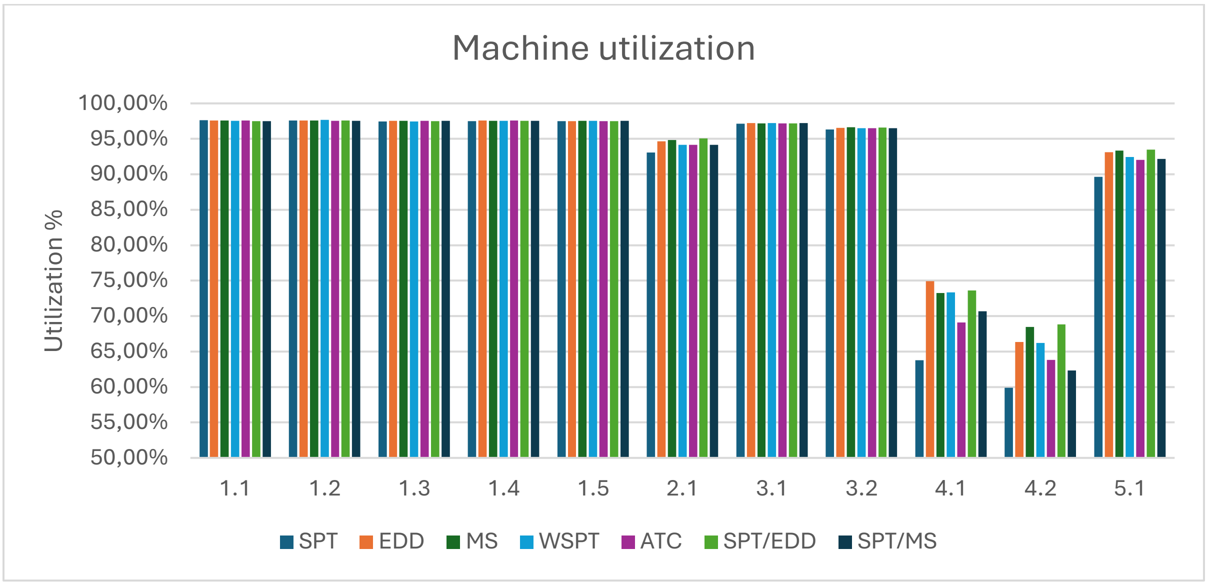

Figure 22: Machine utilization for 5% standard deviation of processing times

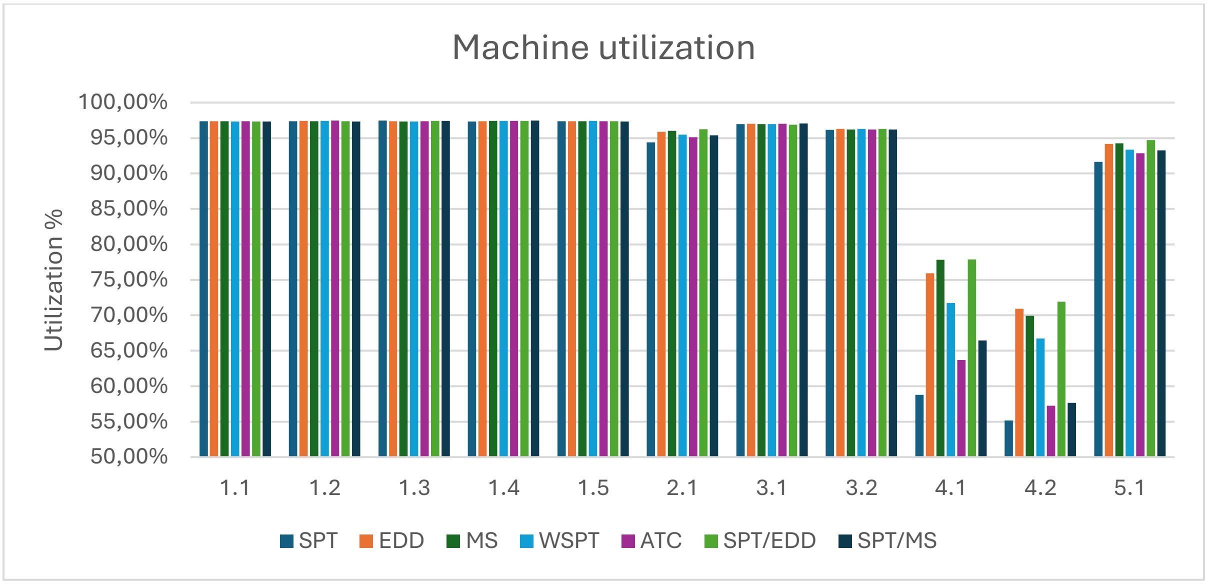

Figure 23: Machine utilization for 15% standard deviation of processing times

9.4. Experiments over- and undercapacity

	4940	5200	5460
SPT	12,62	14,98	13,59
EDD	11,07	14,75	17,80
WSPT	11,62	15,44	18,20
MS	11,75	13,46	12,39
ATC	10,59	12,26	11,01
SPT/EDD	10,95	13,93	12,59
SPT/MS	10,59	12,13	11,17

Table 16: Average lead time for different numbers of orders

	4940	5200	5460
SPT	885,08	1200,90	1091,60
EDD	626,16	1010,52	1309,12
WSPT	658,53	993,52	1091,93

MS	707,01	936,81	911,64
ATC	614,31	870,94	780,69
SPT/EDD	603,25	889,12	771,66
SPT/MS	619,44	837,60	781,83

Table 17: Average in-process inventory for different numbers of orders

	4940	5200	5460
SPT	84,3%	82,6%	83,3%
EDD	94,8%	86,0%	33,9%
WSPT	96,7%	91,8%	56,7%
MS	94,2%	89,7%	79,9%
ATC	96,9%	93,8%	92,3%
SPT/EDD	82,3%	82,0%	84,8%
SPT/MS	96,0%	93,7%	91,1%

Table 18: Due date performance for different numbers of orders

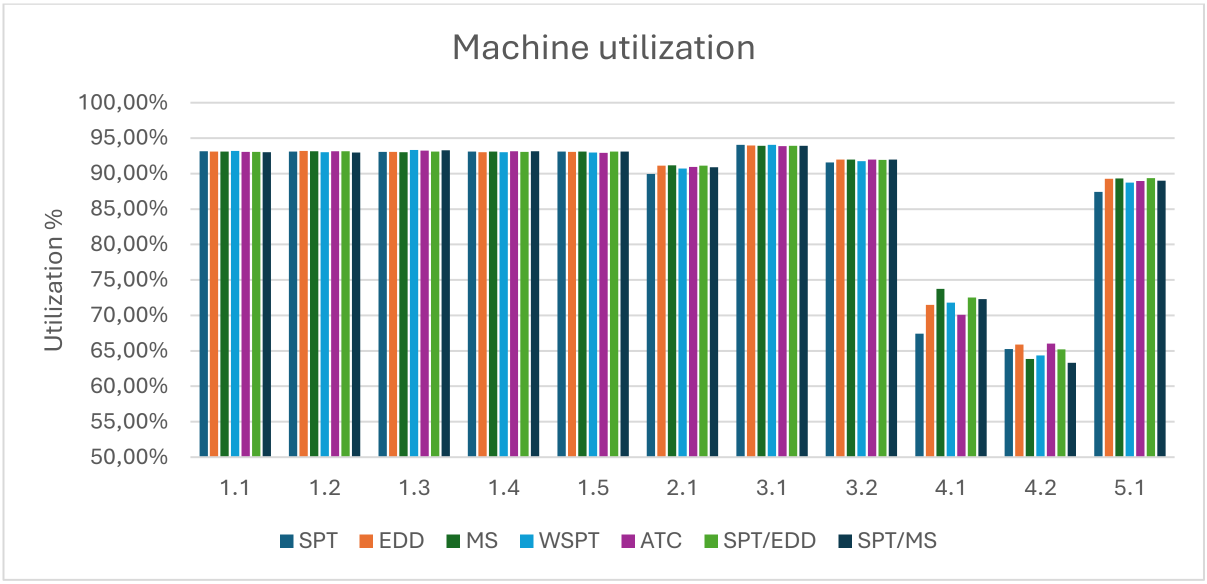

Figure 24: Machine utilization overcapacity

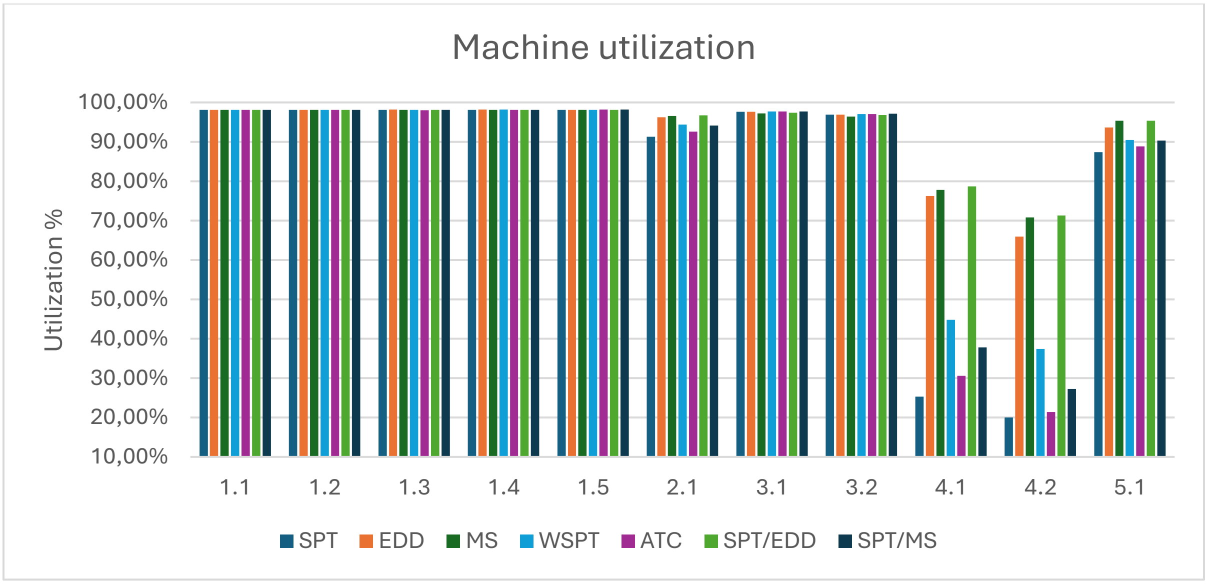

Figure 25: Machine utilization undercapacity

9.5. Experiments with constant order pattern

Dispatching rule	Average lead time (hours)	Average in-process inventory (pieces)	Due date performance (%)
SPT	14,98	1200,90	82,6%
EDD	14,75	1010,52	86,0%
MS	15,44	993,52	91,8%
WSPT	13,46	936,81	89,7%
ATC	12,26	870,94	93,8%
SPT/EDD	13,93	889,12	82,0%
SPT/MS	12,13	837,60	93,7%

Table 19: Results experiment constant order pattern

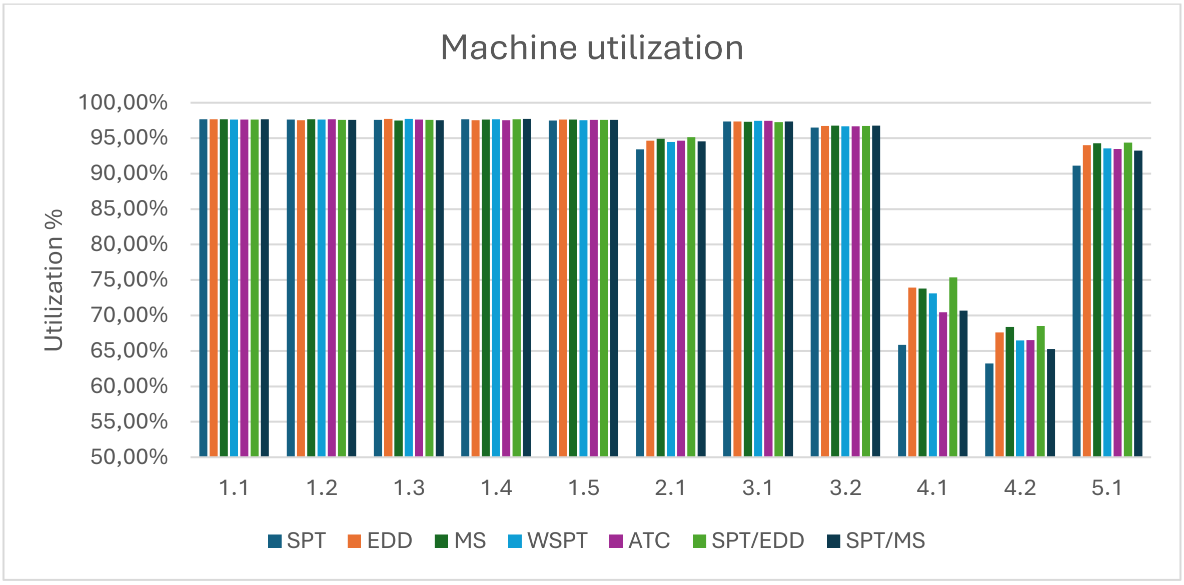

Figure 26: Machine utilization constant order pattern

9.6. Python Code

The Python code that was used in this thesis is confidential and has been omitted from this document.