Improving production planning by flow shop scheduling algorithms

A case study at ForFarmers

Jeroen Teunissen 20-7-2018 University of Twente

GENERAL INFORMATION

University of Twente
Programme Industrial Engineering and Management
Postbus 217
7500 AE Enschede

Title: Improving production planning by flow shop scheduling algorithms
Subtitle: A case study at ForFarmers

Author:
Jeroen Teunissen
Master Industrial Engineering and Management
Production and Logistics Management

Supervisors:
Faculty of Behavioural Management and Social Sciences
Dep. Industrial Engineering and Business Information Systems (IEBIS)
Dr. Ir. J.M.J. Schutten
Dr. P.C. Schuur

ForFarmers Nederland B.V. (Lochem, The Netherlands)
Remco van der Linden

Date of publication: 20-7-2018
First edition
Report, 86 pages
5 appendices, 16 pages

PREFACE

This master thesis is written as part of my graduation project, which I performed at ForFarmers in Deventer to finish my master Industrial Engineering and Management. I grew up in the countryside, which is why I am interested in the agricultural sector. It was interesting and challenging to apply the theoretical knowledge I acquired during my master studies in an agricultural setting.

First, I thank Remco van der Linden for his supervision during my time at ForFarmers. Our weekly meetings helped me to move forward and provided me with new ideas. I thank all the operators for their help in making me familiar with the processes at ForFarmers. Especially, Gerard van Huffelen at whom I could fire all my questions.

Second, I thank Marco Schutten and Peter Schuur of the University of Twente. Their guidance and feedback really improved the quality of this thesis. The new ideas they came up with, made me critically reflect on my own decisions.

I hope you enjoy reading this thesis and that it will be helpful for ForFarmers in using their production capacity more efficiently, or helpful for colleagues who also try to model the processes of a feed mill.

Jeroen Teunissen, July 2018

SUMMARY

"For the Future of Farming," the mission stated by ForFarmers, focusing on long-term success and continuity of the farm. To be ready for the future of farming, ForFarmers wants to know whether their production capacity can be used more efficiently. We investigated whether they can improve their operational production planning by optimizing the order sequence. Besides, we determined the bottlenecks in the process. We conducted our research at the ForFarmers plant in Deventer.

ForFarmers produces compound feed (meal types or pellets) for farmers. The production of compound feed consists of two lines. First, the grind and mix line, which starts with dosing and weighing the ingredients. The larger ingredients are grinded at a hammer mill, after which liquids are added, and all ingredients are mixed at two sequential machines. Meal types are finished after the grind and mix line, pellet type products need to be pelletized at a press line. The product is pre-compacted and pressed into a pellet. Because a lot of heat is generated, it needs to cool down before it can be coated and stored in a finished product silo. Deciding what to produce next is done on intuition of the operator, who needs to have a lot of knowledge of the process and products to produce.

The production process fits the characteristics of a hybrid flow shop (HFS). We model the production process in five sequential stages. Each stage consist of one machine or a group of machines. The first stage consist of the hammer mill, the second stage consist of the first mixer, and the third stage of the second mixer. Stage four represents the press silos and connects the grind and mix line to the press lines. Stage five consist of the press lines. Most stages not only have a processing time, but also an unloading time in which the product flows out of the machine. There is limited or no buffer capacity between stages. So, a job can only move to the succeeding buffer or machine if it is available. Jobs are combined in large batches of the same product at the press lines. There are five, unrelated parallel press lines. Process times differ per press line.

To optimize the production sequence, we design our own algorithm. We use a divide-and-conquer strategy that divides the problem in two subproblems. In the first subproblem, we optimize the press lines with use of adaptive search. We added an additional rule to the standard adaptive search algorithm that schedules all jobs of the same product after the first job of the product when it is chosen by adaptive search. For each job, when we schedule it at the press, we immediately assign it to a press silo. Thereby, we determine a release date and due date for the grind and mix line, such that the job is produced in time and fits in the press silo when it finishes processing at the grind and mix line. Next, we solve the second subproblem with use of simulated annealing. Given the best schedule at the press lines, we try to find a sequence for the grind and mix line that optimizes the overall objective value. Therefore, for each grind and mix line sequence that we evaluate, we have to change the press schedule such that no job starts at the press before it finishes at the grind and mix line. When the schedule is feasible, we evaluate the objective value, according to the simulated annealing procedure.

We use a multi-objective optimization that consists of two parts. First, we optimize the maximum tardiness, total tardiness, or the number of jobs that is late. Given the best tardiness solution value we find, we optimize the make span, the total make span of all machines, or the total make span of the press lines plus the flow time per job per machine of the grind and mix line. We define six test sets. Except for one test set, for each test set a zero-tardiness schedule is found. Therefore, to determine the performance of our algorithm, we use the three make span related objectives.

ForFarmers can improve the production planning by using an algorithm to optimize the sequence of jobs for production. The extent of improvement differs per objective per test set. For the make span objective, we found an average improvement of 8.2%, for the total make span objective an average improvement of 2.6%, and for the total make span press lines plus flow time grind and mix line an average improvement of 4.8%. At the press lines only, the total make span can be improved by 5.1%.

The press lines are the bottleneck for ForFarmers in Deventer. One grind and mix line is able to supply five press lines and to produce the meal products.

The most important recommendations and suggestions for further research are:

- Our research shows that efficiency increases by scheduling with our scheduling algorithm. We recommend ForFarmers to further investigate how it can be implemented to advise operators in daily production planning. The tool we programmed and used for our analysis has its limitations for daily use. Therefore, we recommend ForFarmers to find a software developer that can built a professional tool, which supports operators and is user friendly.
- At the press lines, we save time by batching jobs of the same product to reduce the number of times a cooling machine needs to empty, because the next job is of a different product.
- At the grind and mix line, we save time by clustering jobs of the same product because there is a setup time in stage 3 if the job is of another product as the preceding job.
- We suggest to investigate how the transport to and storage in finished product silos can be modelled/included. Resources are limited and not each best schedule found is feasible if we take the storage into account.
- We also suggest to investigate how to add the dosing and weighing systems to the model. Especially when the situation changes because of investments or other ingredients; or when the model is implemented in other plants. If the dosing and weighing systems are for some products the machine with the largest processing time at the grind and mix line, they need to be included because depending on the sequence, succeeding stages can have to wait until a job finishes dosing and weighing.
- Our algorithm only uses one press silo for jobs of the same product. It can be extended by adding control rules that can use both silos for jobs of one product. The assumption of using only 1 silo causes larger make spans at the machines of the grind and mix line for the optimized schedules, compared to the original schedule. Hence, larger improvements can be found if both silos can be used for jobs of one product.

ABBREVIATIONS

ACO	Ant colony optimization
AIS	Artificial immune systems
B&B	Branch and Bound
GA	Genetic algorithms
HFS	Hybrid Flow Shop
MIP	Mixed Integer Programming
MS	Make span objective
MSPP	Machine Scheduling Problem in Production
MTS	Make-to-stock
MTO	Make-to-order
NN	Neural networks
SA	Simulated annealing
SBP	Shifting bottleneck procedure
TMS	Total make span objective
TMSPFT	Total make span press plus flow time jobs at the grind and mix line objective
TS	Tabu search

TABLE OF CONTENTS

1 INTRODUCTION

In the framework of completing my masters study Industrial Engineering and Management at the University of Twente, I perform research at the plant of ForFarmers in Deventer. This thesis focuses on the operational production planning, trying to optimize the sequence of orders for production. Section 1.1 introduces the company, Section 1.2 identifies the core problem, Section 1.3 explains the research design, and Section 1.4 lists the research deliverables.

1.1 COMPANY INTRODUCTION

ForFarmers is a conglomerate of feed mills and trade companies. They operate in The Netherlands, Belgium, Germany, the United Kingdom, and Poland. In The Netherlands, there are 9 production locations, mostly in the eastern part. Each plant is dedicated to one type of animal, except for one location, which produces Bio Feed for different species.

ForFarmers' mission is stated as: "For the Future of Farming". They collaborate with farmers to work on long-term success, by focusing on the continuity of the farm and a financially healthy sector. They sell 9.3 million tons of animal feed per year with a revenue of € 2.1 billion in 2016, which makes them market leader in Europe. Their strategy states, among others, that ForFarmers focuses on autonomous growth and extension in Europe and adjacent regions. They want to become an important player in each market they enter. Therefore, there is also a focus on making their systems and processes more efficient and using the same procedures in each plant.

The history of ForFarmers starts at the start of the 20th century. Local cooperatives of farmers existed, which worked together and merged during the years into larger cooperatives. In 1989, the company ABC U.A. emerged which merged in 2000 with CTA, into ABCTA. In 2006, they expanded their activities to Germany and Belgium and they changed their name into ForFarmers. In 2011, Hendrix was taken over, which enlarged their market share and their number of production locations in The Netherlands. In 2012, ForFarmers took over the British animal feed producer BOCM PAULS. Thereby attaining 20% market share in Great-Britain and 11 plants. The last takeover was 'Vleuten-Steijn Voeders', in 2016. This increased their market share in the south-eastern part of The Netherlands. In 2018, they started a joint-venture with Tasomix in Poland. Figure 1.1 shows the development of the name ForFarmers.

FIGURE 1.1: TIMELINE FORFARMERS (ARTICA, 2018)

Since 2006, the number of employees of ForFarmers increased from 546 to 2273 in 2016. They have grown a lot during the last decade and they aim to continue this during the forthcoming years. Figure 1.2 shows the locations of ForFarmers in the countries mentioned above.

FIGURE 1.2: LOCATIONS FORFARMERS

ForFarmers has 9 production locations in The Netherlands. They are very similar in terms of processes used, mainly meal types and pellets are produced. We divide the process at ForFarmers into five main subprocesses. First, ingredients are stored, mostly in silos. Next, by grinding and mixing processes, meal is produced. This could be stored directly in silos or it could be pelletized. Pellets are also stored in silos. Finally, trucks are loaded, and the final product is transported to the farmer. We introduce each sub process briefly in this section. Section 2.1 elaborates more on the production processes.

Most ingredients, e.g. maize, wheat, barley, and soy, are delivered by barge. Large cranes are used to empty the barges, after which the ingredients are stored in silos. Other ingredients, e.g. minerals and some ingredients for specialty feed, are delivered by truck. These ingredients are stored in silos or in a conventional warehouse, in case of bagged goods.

Production starts with dosing and weighing ingredients. Minerals are directly going to a mixing stage. Before other ingredients move to the mixing stage, they are sieved first, after which the large ingredients are grinded in the hammer mill. After mixing, meal types are finished and are stored in silos; pellets need to be pressed. This second phase of production starts with a pre-compaction process, after which the mix is pelletized. Because a lot of heat is needed for this process, pellets need to cool down before they are coated and stored.

Based on their schedule, and whether all products are produced, the trucks are loaded. Trucks consist of different compartments, in which various products can be transported. There is a loading street, through which trucks can drive to be loaded. Above the loading street, there are small silos in line, which contain at most the volume of one compartment of the truck. These small silos are filled before the truck arrives, such that the truck does not have to wait unnecessarily.

Finally, the ordered goods are transported to the customer, mostly by own trucks. Figure 1.3 shows a process flowchart of the processes described above. Because the sequence of operations is the same for each product, we classify the process at ForFarmers as a flow shop (Graham, Lawler, Lenstra, & Rinnooy Kan, 1979). ForFarmers operates under a hybrid MTO-MTS strategy (Soman, van Donk, & Gaalman, 2004), i.e., fast movers or general products are produced make-to-stock (MTS), slow movers or specialties are produced make-to-order (MTO). Fast movers are products that are sold most, slow movers are sold less often.

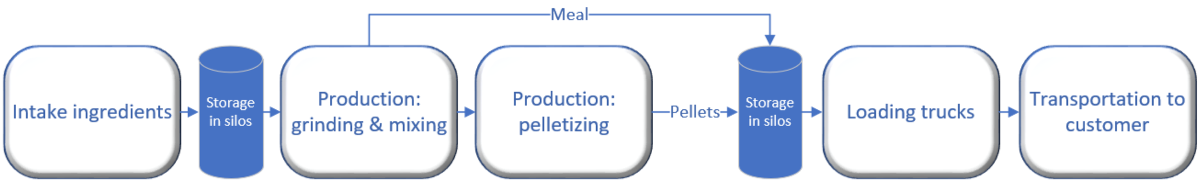

FIGURE 1.3: GENERAL PROCESS FLOWCHART

1.2 Problem Identification

Section 1.1 states that ForFarmers increased their market share and production capacity during the last decade and are planning to grow further the forthcoming years. To be able to do so, many operational and logistical processes have to be improved or optimized through mostly continuous improvement initiatives. An example is production planning; ForFarmers has the feeling that they do not have sufficient insight in which machines or workstations are a bottleneck and whether the production could be scheduled more efficiently. The goal of this research is to provide insight in these bottlenecks, before and after optimizing the production schedule. Exact order deadlines are becoming available after the transport planner has scheduled their transport. Before, only a range of days is known to operation, in which the order should be delivered. Orders for make-to-stock products are created when more is ordered than is currently in stock or when there is time available in production. ForFarmers does not use a reorder point. The planning horizon has a length of about two days. Therefore, the problem can be classified as an operational planning problem (Giebels, 2000).

To identify the core problem, often a problem cluster is made (Heerkens & van Winden, 2012). However, in this project, the core problem follows from the problem stated by ForFarmers. More insight in bottlenecks is needed and a new scheduling strategy needs to be investigated and modelled, to make the production more efficient. Therefore, the core problem is stated as:

The available production capacity is not used efficiently enough

An important note on the core problem is how to measure the efficiency. There are two variables, often used to measure the efficiency of a production schedule: the makespan and the lateness. The makespan is the time the last job is finished; the lateness is the difference between the time a job is finished and its due date. The lateness value could also be negative. Instead of lateness, tardiness is also possible. The tardiness is the positive lateness, the maximum between the lateness and zero.

We think the lateness/tardiness variable to be more appropriate in case of ForFarmers. After transport is scheduled, the order needs to be finished in time. In case of a replenishment order, the stock needs to be replenished in time. If we would use makespan as variable, we want all products to be finished as early as possible. Thereby, utilizing the machines most efficiently. However, because transport is already scheduled, finishing an order early has no added value. On the contrary, it must be stored for a longer period, for which more storage capacity is needed.

1.3 Research Design

The problem as stated in Section 1.2 is quite complex. Therefore, it is important to think carefully about what to include or not. Section 1.3.1 describes the scope. Next, Section 1.3.2 describes the research problem and questions, and the approach to answer them.

1.3.1 Scope

There are differences in complexity between production locations. For example, the number of machines for each stage of the process, but also the availability of data differs. Because each location is specialized in one animal type, there are also differences in products produced per location. To reduce the complexity over locations, first the location in Deventer is investigated, which produces feed for cows. Deventer has a lot of data available and is less complex. There are fewer machines in Deventer and there are fewer products produced than in e.g. Lochem. Still, there are 256 products in their assortment, of which only 9 are responsible for 50% of total turnover, and there are workstations with multiple machines.

Efficiency will be improved by optimizing the sequence in which a given set of orders is processed. Figure 1.4 shows the sub processes explained in Section 1.1, a dashed rectangle indicates the processes included in the scope. These steps are needed to transform raw materials into compound feed, such as pellets or meal. Before production is started, raw materials are stored. This process is not taken into account, we assume raw materials to be always available for production. After production, the finished product is stored temporarily until it is loaded for transport and transported to the customer. The process of storing these products in silos is not taken into account. Although the resources needed are limited, which can influence the feasibility of a solution. However, modelling the storage is rather complex and does not fit in the time schedule of this research. The loading of trucks and transportation to the customer are also not taken into account.

Of the grind & mix stage, we do not take the dosing and weighing process into account when modelling the process. We explain this decision in Section 4.1. We still think it is important for the reader to know something about the complexity of the dosing and weighing process. Therefore, we explain this process, the corresponding times and the current performance in respectively Section 2.1, 2.4 and 2.5. We also include our findings for the dosing and weighing systems for readers who are also trying to model the feed mill processes.

Hence, we focus on the location in Deventer, taking only the production steps without the dosing and weighing systems into account. Section 2.1 elaborates more on the process flows of production.

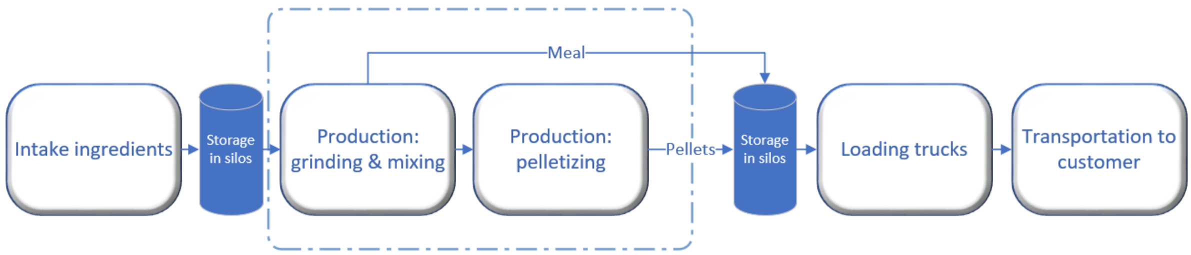

FIGURE 1.4: PRODUCTION STEPS IN SCOPE

1.3.2 Research Problem

This section states the research problem and questions, and a brief explanation of the problem approach per question. We summarize all knowledge, which we want to obtain, in two research problems. The first research problem is:

How can ForFarmers Deventer improve their operational production planning by optimizing the order sequence such that tardiness is minimized?

This research problem mainly focuses on modelling the problem and designing an algorithm to solve it. The second research problem is:

What are the bottlenecks in the process, before and after optimizing the production planning?

Information on bottlenecks in the process is important to make well substantiated investment decisions. Interesting to know is which bottlenecks are taken away by smarter planning of the production.

To answer the research problems, we state several research questions below. We think they are useful to provide an answer to the research problems stated.

Research Question 1: *How is the production and production planning currently organized and what is the current performance?*

1.1. *What does the process flow look like?*
1.2. *What is the current way of planning?*
1.3. *What are constraints that need to be considered?*
1.4. *What are the processing, setup and removal times?*
1.5. *What is the performance and are the bottlenecks for the current planning strategy?*

In Chapter 2, we describe the context. To be able to determine the current and improved performance, it is important to know what the process looks like. For the current performance, the current way of planning is important. Constraints are important to know, to be able to find a feasible solution.

To be able to test the performance of a new planning algorithm, we determine the performance of the current planning approach. To do this, we analyse process data to obtain processing times and setup times. We identify bottlenecks based on historical production data, using a sample of four different weeks.

Research Question 2: *What is currently known in literature on machine scheduling for flow shops?*

2.1. *How could a machine scheduling problem be classified?*
2.2. *Which algorithms are currently known to schedule machines in flow shops?*

In Chapter 3, we provide an overview of existing literature on machine scheduling for flow shops. First, we explain how machine scheduling problems are classified. Next, we give examples of algorithms known in literature for flow shop problems.

Research Question 3: *How do we find an improved production schedule for ForFarmers?*

3.1. *How do we model the production planning at ForFarmers?*
3.2. *What approach/algorithm should we use to find an improved schedule?*

Before applying scheduling algorithms to find a solution, we model the production planning in Chapter 4. We define stages that consist of one machine or a group of machines. Next, we explain the algorithm we propose to find an improved schedule. This algorithm is partially based on algorithms from literature and is used for experiments and analyses in Chapter 5. Therefore, we use professional programming software, such as Embarcadero Delphi, to program the algorithm.

Research Question 4: *What is the effect of the planning algorithm on the performance?*

In Chapter 5, we analyse the performance of the implemented planning algorithm. We compare the new performance to the current performance to be able to express the benefits. We perform an additional analysis for the grind and mix line only to analyse the effect of different objectives on the schedule at this line. Next, we do some sensitivity analysis on the parameters of our algorithm. Finally, we test our model on feasibility with respect to the storage in and transport to the finished product silos.

Finally, we present our conclusions and recommendations in Chapter 6.

1.4 Research Deliverables

The deliverables of this research are:

- Insights in the processing times at the different machines of the production process. Whether processing times depend on quantity and/or product to produce and the duration itself. Insights in which line can best be used to pelletize a product the fastest.
- Insights in the utilization of machines and the bottlenecks in the process. Especially whether the bottleneck changes after scheduling production by an algorithm.
- A dedicated algorithm designed for feed mills to improve the production planning. Programmed to be a tool that can be used to make scheduling decisions for a set of products to produce and for our own analyses.
- An answer to the question whether scheduling production by an algorithm can improve efficiency and how much time we can save.
- Recommendations based on general rules that follow from our analysis.
- Recommendations for the transport and storage in finished product silos, which are not included in the scope of the scheduling algorithm.

2 Context Analysis

This chapter focuses on the first research question: *How is the production and production planning currently organized and what is the current performance?* To answer this question, Section 2.1 explains the process and its flows. Section 2.2 elaborates on the current way of planning and Section 2.3 explains some constraints of the process that we think to be important to consider for production planning. Section 2.4 is about processing and setup times and provides some more in-depth insights in the process description of Section 2.1. Section 2.5 provides a view on the current performance and the bottlenecks currently present in the system. Finally, Section 2.6 gives the most important conclusions from this chapter.

2.1 Process definition

This section explains the production processes taken into account in more detail. As Section 1.1 indicates, the production consists of two phases: the grind and mix phase and the pelletizing phase. After pelletizing, the storage process takes place. Figure 2.2 shows a process flowchart of the production process, which we explain further in this section. The blue processes are of the grind and mix phase. The red processes are part of the pelletizing phase. The plant of ForFarmers in Deventer consists of seven floors, which are indicated by horizontal lines in the figure. The ingredients/product moves several times up and down through the plant. Often, the next stage is below the previous one, using gravity to move the product.

Grinding and mixing

This part focuses on the blue processes in Figure 2.2. The production starts with dosing and weighing the ingredients at the ground floor, such that the right amount will be in the product. This process is automatically done by three dosing and weighing systems (DW): DW1, DW2, and DW3. DW1 is used for minerals, DW2 and DW3 for other ingredients. They can only process one ingredient at a time and each dosing and weighing system has its own ingredient silos. These silos are positioned right above the machine. The ingredients are stored temporarily in a buffer, buffers are indicated with the abbreviation BU in Figure 2.2. Buffer 1 collects all minerals for a job, buffer 2 collects all other ingredients from DW2 and DW3. DW2 and DW3 need to work on the same job, because they share the same succeeding buffer. Before the minerals of DW1 can be mixed with the larger ingredients of DW2 and DW3, the larger ingredients often need to be pre-processed. They are grinded on a hammer mill to get finer particles. Therefore, they are transported to the upper floor by an elevator transporting system. This system consists of boxes connected to a chain which goes around, see Figure 2.1. Minerals are light and do not need to be transported to the upper floor, see Figure 2.2. Therefore, they are blown into the first mixing machine.

Figure 2.1: Elevator transporting system (Arvos-Group, 2018)

At the upper floor, the larger ingredients are stored in alternately buffer 3 or buffer 4. In case they do not need to be pre-processed before mixing (this is known by the system), they can surpass the hammer mill machine and enter the buffer in front of the first mixing machine (BU6). Often, they are too rough, and they need to be grinded on the hammer mill first. Before starting this process, they are sieved to speed up the process. The small parts are directly going to buffer 6; the big particles go, via buffer 5, to the hammer mill. At the hammer mill, the grinding is done by two machines simultaneously. The volume of a job is divided over the machines to speed up the process. Figure 2.3 shows an example of a hammer mill. Hammers are rotating and grind the product against a screen. After grinding, the fine particles fall into buffer 6.

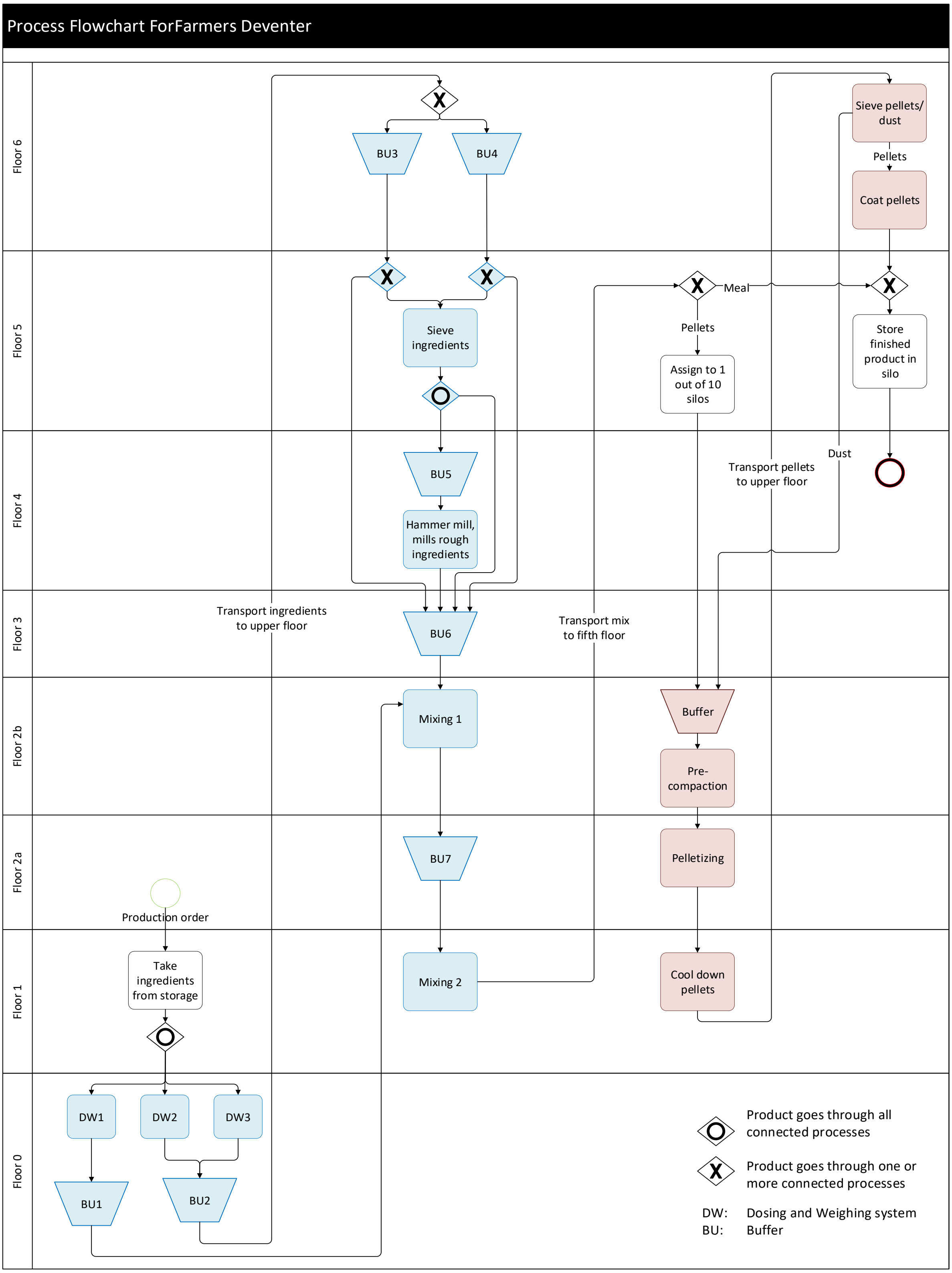

FIGURE 2.2: PROCESS FLOWCHART PRODUCTION PROCESS. BLUE = GRIND AND MIX LINE, RED = PRESS LINE

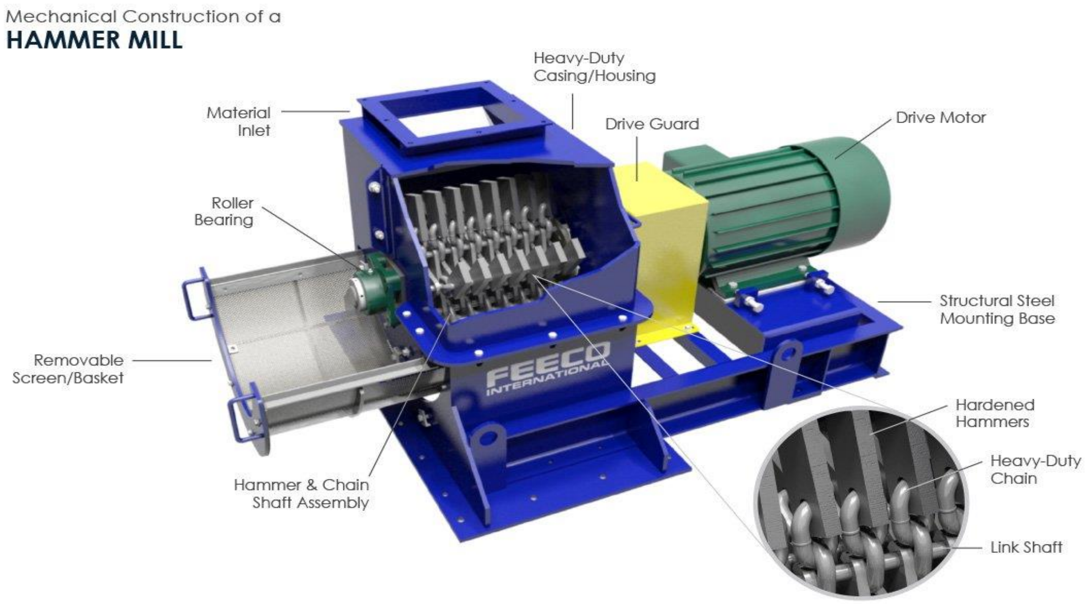

FIGURE 2.3: HAMMER MILL (FEECO INTERNATIONAL , 2018)

Mixers mix liquids with the ingredients in two mixing stages. In the first mixing stage, the minerals and grinded ingredients are mixed and possible liquids are added. The mixer mixes a certain amount of time before it unloads the product. The product then goes to buffer 7, after which it moves via a second mixer to an elevator transporting system that transports it to the next stage. In this second mixing stage, the liquids are added while the product moves through it.

Pelletizing

The next part focuses on the red processes in Figure 2.2. After grinding and mixing, the product is moved to the fifth floor by an elevator transporting system. The meal products are stored in a silo. The pellet products are stored in a silo above a press line. There are 5 press lines that each have 2 dedicated silos as predecessor. Figure 2.4 shows a process flowchart of one of these five press lines, starting with the dedicated silos. The process flow at the other press lines is the same. Multiple jobs on the grind and mix line can be combined in one large batch before pelletizing, if they are of the same product. One silo is in use during pelletizing, which is a continuous process. The next product can already be produced at the grind and mix line and stored in the second silo. Thereby, the press line can continue faster with the next product when a batch is finished. To avoid contamination, it could be needed to clean the press line in-between two jobs. However, the plant in Deventer does not have products with contamination issues. There are two products, for which another mould is needed. A mould is used to produce pellets of a certain diameter. All other products are produced on the same mould and hence do have the same diameter size. So, we identified two possible kinds of setups, of which only one is applicable to Deventer.

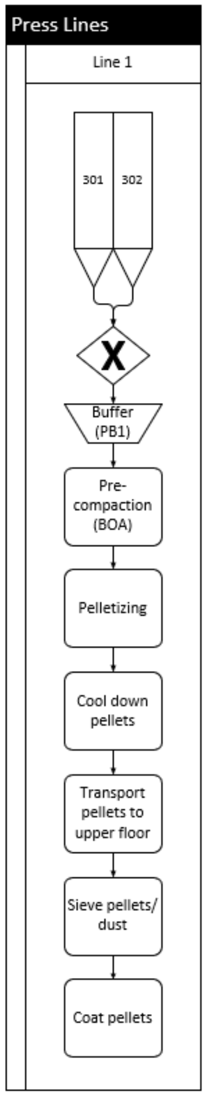

FIGURE 2.4: PROCESS FLOW CHART PRESS LINES

The pressing of pellets starts with a pre-compaction process. Steam is added to the product and it is pushed through a narrow gap. Therefore, the first three lines use a so-called BOA, the last two use an expander. The latter one can produce at a higher speed. However, some products require a BOA to attain their required quality. Figure 2.5 shows a BOA, which consists of a mixer and rolls that push the product through a gap.

FIGURE 2.5: BOA (ALMEX EXTRUSION TECHNIQUES, 2018)

Figure 2.6 shows an example of an expander. It uses a screw to push the product through a narrow, annular gap. There are products with a preference to be processed on a line with a BOA or with an expander. There are also other differences between the lines with a BOA, e.g. the thickness of the mould of the press and the maximum possible power. Before pre-compaction, again some liquids are added. After pre-compaction, the pellets are pressed. The product is pressed through a mould. During pelletizing, a lot of heat is generated. Therefore, the pellets need to cool down before they are stored. Each press line has its own cooling machine. The pellets are cooled with air from outside the factory. The temperature of the pellets may not differ more than 10 degrees Celsius with the outside temperature, otherwise they will clump together. Only after the cooling machine is finished and empty, the next job can start on the press line.

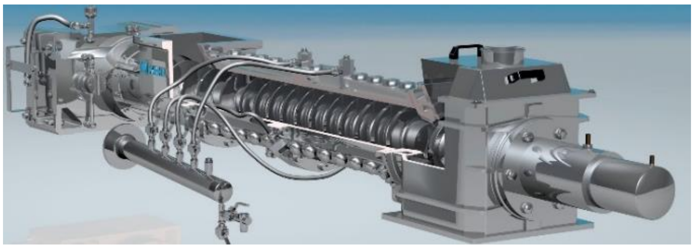

FIGURE 2.6: EXPANDER (DIRECTINDUSTRY, 2018)

After cooling down, the pellets are transported to the upper floor. There they are sieved, after which the dust is returned to the pre-compaction stage to be reprocessed. If pre-compaction of the current job is already finished, the dust is stored in a silo. Tiny amounts of leftovers can be re-used as extra addition in other products. The pellets are coated and then transported to a silo to be stored. The press line is a continuous process in which each machine works on the same product simultaneously. It only takes some time before the first pellets enter each machine. When the last pellets of a product are cooled down, the next product can start at the pre-compactor. The last pellets are coated and transported to their silo.

Transportation to silos finished product

The transportation to the silos is quite complex. From the first pellet to be coated until the last, the press line requires one or two chains to reach the silo. Figure 2.7 shows an example of a transport chain. During the period a press line uses a chain, the resource cannot be used by other press lines. There are 6 primary chains, one per press line and one that delivers meal from the grind and mix line. They are denoted by the abbreviation PKx or MKx. PK1 and PK5 can deliver directly to some silos, e.g. PK1 to silo 417. To reach most silos, a second chain is needed. There are 5 secondary chains, PK6 to PK9, and MK2. They all deliver to a different set of silos and are delivered from a different set of primary chains.

Figure 2.8 shows some primary and secondary chains, and some silos to illustrate the relations. Each arrow indicates that a product can move in that direction from one chain to another or from the chain to a silo. In total there are 35 silos for finished products, of which 27 for MTS products and only 8 for MTO products. There are 14 pairs of silos of which only one can be reached from PK6 or PK7 simultaneously, e.g. if the product of silo 409 is produced, the product of silo 410 cannot be produced because the silo cannot be reached. In case of e.g. silo 413 and 414 this does not hold, the other silo can be reached by PK9, but only if PK9 is not used for another product. Appendix A shows a table that contains all connections between silos and belts. It is possible to trace back which press lines can supply which silos via which belts.

FIGURE 2.7: TRANSPORT CHAIN (ALIBABA, 2018)

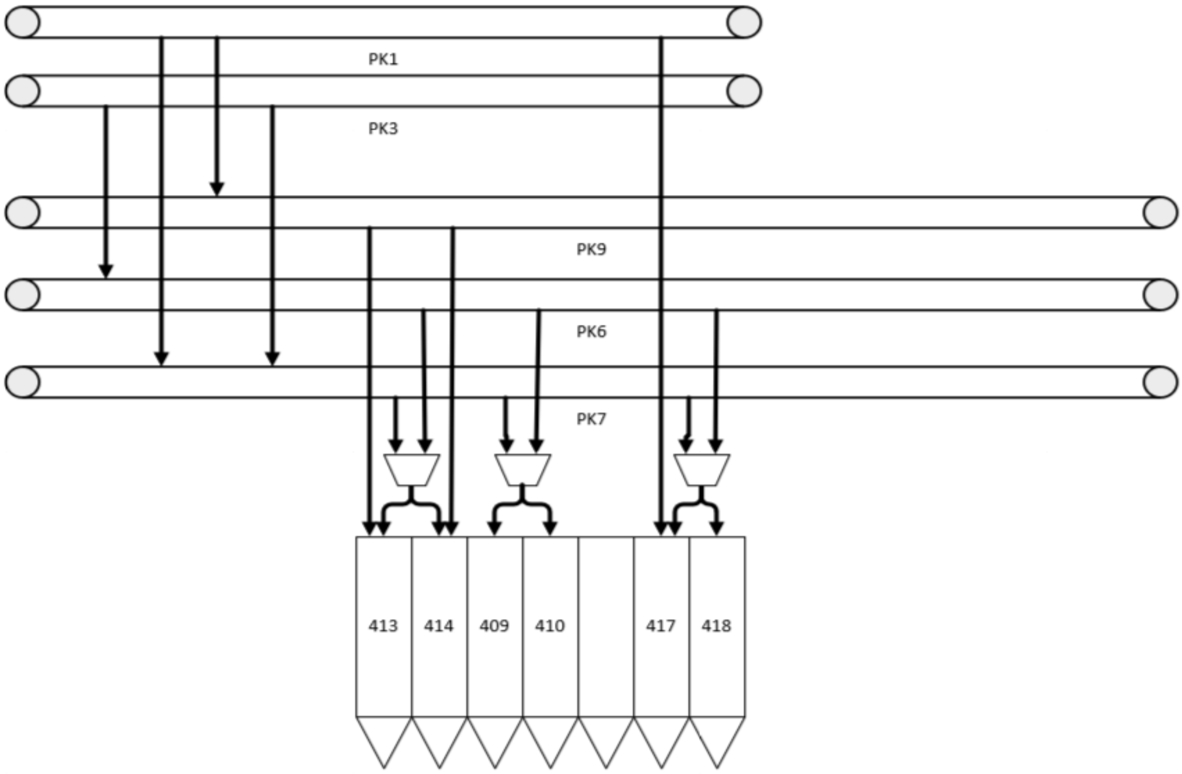

FIGURE 2.8: EXAMPLES OF CONNECTIONS BETWEEN PRESS LINES AND SILOS.

The production process described above is automated by use of a Manufacturing Execution System (MES). The MES software used is ISA 95 based and an integration of the traditional MES and the lower level PLC's that control the machines. It is specialised software for feed mills. All machines, conveyors and silos are centrally controlled in a control room.

2.2 CURRENT WAY OF PLANNING

Planning is in the current situation a dynamic process that relies upon the intuition and experience of the operators. There is a lot of communication between the transport planner, production operators, and the truck loading operator. The transport planner verifies with production planning whether transport can take place in time. Sometimes the production operators indicate that production of certain products could not be finished in time for transport. They notify the transport planner, who then delays or reschedules transport.

Jobs for production can be divided into make-to-order (MTO) and into make-to-stock (MTS). Most fast movers are MTS. They have their own storage silos and are produced when ordered demand is higher than the current stock level, or when there is unused capacity on a press line. On the other hand, slow movers, often special products, are produced MTO. There are some flexible silos where they can be stored temporarily. Most of production is MTS. On the grind and mix line, 60% of the jobs are MTS. This is 71% of the total weight produced and 63% of the total production time. Because meal products are MTO and they are not processed on a press line, the ratio of MTS orders on the press lines is even higher.

It is hard to identify a clear, structured approach of scheduling production. Still, we identify some important general rules. There is limited storage capacity for meal and MTO pellets, see Section 2.3. This forces the operators to plan in a certain way. First, there are two silos that are dedicated to store meal products. Because there is a large variety of meal products, these are MTO. Because only two silos are available, it is not possible to produce much meal products in advance. Truck drivers, who deliver meal to the customer, are asked to call one hour before they arrive at the plant. In that hour, production schedules the jobs for that truck. This approach results in a last-minute change in the schedule, because other jobs are delayed and first the meal is produced.

Second, MTO jobs are mostly scheduled during the night and morning, because there are only 6 silos available to store them for a larger amount of time. During the night, trucks are not used. The night operators produce all MTO jobs with a due date in the morning, during the night. Instead of storing these products in the silos, they already fill the trucks in advance. So, the driver can depart immediately in the morning. During the morning, orders are produced that need to be delivered in the afternoon. Because of the small amount of storage capacity, these jobs are mostly scheduled close to their due date for transport. Another approach used to deal with the low amount of storage capacity is to store lesser amounts of MTO jobs temporarily in wooden chests. In one chest, 2000 kg is stored. The chests later need to be emptied in a truck.

MTS products are more produced during the afternoon and evening, or during the night and morning when there is capacity left or when an out of stock situation occurs. There is no reorder point used. The current volume in stock is compared with the volume ordered. If there is more ordered than available, a production order is created to restock the product to its maximum amount. When there are no shortages, the operator chooses a product of which a substantial amount could be produced. The product he chooses, depends on his experience.

Some products need to be processed on a line with an expander, lines 4 and 5, to attain the required quality. For other products, a line with a BOA is more fit, lines 1, 2 and 3. Most MTO pellets are pressed on line 1 or 3; line 1 has a direct connection to MTO silos, so this saves the use of one resource. The lines with an expander produce almost only MTS products, because they are needed in large quantities and an expander can produce faster. Some products are difficult to pelletize; therefore line 2 is often selected, because the mould in line 2 is thicker. All these preferences are considered when scheduling the next order(s) to produce.

The job size on the grind and mix line is limited to 6000 kg, because of the maximum amount that the first mixer can contain. Often more is needed, and multiple jobs are created for this product, especially in case of MTS products. These jobs are combined to one large batch on the press line, which saves time because it takes some time to reach the desired throughput speed. Figure 2.9 illustrates the batching for the press line with an example of a product at line 5. This product is highlighted to indicate when it is processed at which machine. One large batch is processed at line 5, for which in this case 4 times 5 batches of 6000 kg are produced at the grind and mix line. Not all these batches can be produced at once, because both silos above the press line can only contain 30,000 kg.

FIGURE 2.9: PART OF GANTT-CHART ILLUSTRATING THE BATCHING ON THE PRESS LINE. (HM = HAMMER MILL, M1 = MIXING 1, BU7 = BUFFER 7)

2.3 Constraints for production planning

The process is quite complex and there is limited equipment available for processing, transporting, and intermediate storage of the product. As a result, there are constraints for the production planning. Important constraints are:

- Storage capacity of finished meal products is limited. There are only two silos available for storing meal products. This results in producing meal not much before transport is scheduled. So, there is not that much freedom in where to fit these orders into the schedule.
- Storage capacity of make-to-order products is limited. There are only six silos available for storing MTO products. Consequently, MTO products can only be produced close to the moment of transportation, such that less inventory capacity is needed.
- Not all MTS pellets can be pressed simultaneously because they require the same conveyor belt to reach their silo. Section 2.1 explains the complexity in these relations between belts and silos.
- Between most stages in the mill and mix phase of production, there is only one buffer. Hence, only one job can be stored temporarily. If the next job is finished too early, it must wait in the machine. Thereby, the job is blocking the (preceding) processes.
- Not each product could be pelletized on each press line, because of quality issues and configuration of the different lines.

2.4 Processing, setup, and removal times

This section is about the processing, setup, and removal times. We first introduce different kinds of setup times that are possible, our approach to determine process times, and the data we use. Next, we look at the different stages of the process and their process times. We give some more in-depth insight in the process of these stages, compared to the general description in Section 2.1. We look consecutively at the dosing and weighing stage, the hammer mill, the first mixing stage, the second mixing stage, and the press lines.

The MES keeps track of all kind of information per job. Data as the start time, stop time, running time, and weight produced are available per job per stage of the process, of the last year. We analysed this data, to determine the duration of the processes at the different machines. The durations are input for our model, which is used to experiment with the order of the jobs. Important to note is that besides process and setup times, there are also removal times. Most stages have a so-called 'LOS' time, which comes from the Dutch word 'lossen', which means unloading. The 'LOS' time is the time it takes to empty the machine, such that a new product can be produced.

There are several kinds of setup times possible in the animal feed industry. On the hammer mill, different sieve sizes can be used per product; so dependent on the sequence, the sieve needs to be exchanged. However, at ForFarmers in Deventer, all products require the same sieve size. The press has a mould, which provides pellets of a certain diameter. Different products can have a different diameter; so dependent on the sequence, a changeover is required. In Deventer, there are 2 products with a diameter of 8 mm. The other products have a diameter of 5 mm; so, the number of changeovers is small. To decrease the number of changeovers, the 8 mm pellets are produced MTS, in large amounts. A third type of setup time is due to contamination issues. Some products cannot be produced after each other because ingredients, which may be left behind in the machine, are not allowed in the next product. To still produce them after each other, the machine needs to be rinsed.

Although the process is fully automated, the durations of each stage of the process are not known. The time a job enters and leaves a machine is logged, so we can calculate the duration of a single job at each stage. This duration does not include changeover time. Changeovers or rinses are done before the job enters the machine. Because only the start and stop time are logged, possible failures increase the duration. Also, if succeeding stages are blocked because a machine is failed, or the process takes more time, the duration of the current stage increases. Furthermore, the duration can depend on the quantity and/or product to produce.

To determine the process and removal times, we use a statistical approach to detect correlations and to remove outliers. First, we create three types of graphs: a histogram, boxplot, and scatterplot. The latter one shows whether there is a relation between the duration and the quantity produced. The histogram and boxplot show outliers. If we suspect a relation between the duration and the product, we create scatter plots to confirm this. However, the duration can also depend on both product and quantity. We group the durations per product and/or quantity and we rounded the quantities per 100 kg to get an acceptable data density per group. Within a group, we analyse the relation between the duration and product, by creating new scatterplots. When there is no relation with quantity or product, outliers are deleted based on the whiskers of a boxplot. Hence, data points outside 1.5 IQR (interquartile range) of the lower and upper quartile are deleted. If data is grouped on quantity and/or product, the boxplot analysis is automated. We calculate per group the boxplot 'whisker' values and delete the outliers.

We take the median values as estimations for the process time of the groups per quantity. The median values are more robust in case of outliers. The data density differs per group. Some groups consist of a lot of data points, others of just a few. Especially in smaller groups, outliers can remain undetected by our approach, influencing the average value too much. We could create a higher data density by creating fewer groups, with a larger quantity range. However, this would also increase the variation in the durations per group. Therefore, we choose to group the data per 100 kg and to use the median values.

We use data from 2-2-2017 to 3-3-2018 for our analysis. Detailed data is not saved for much more than a year. Nevertheless, one year of data consists of more than 50,000 jobs on the grind and mix line and 9,500 on the press lines. In October 2017, the hammer mill of the grind and mix line got an upgrade. The capacity is increased, and processing times are changed. Therefore, we only take the data from 20-10-2017 to 3-3-2018 into account for the grind and mix line.

Dosing and weighing

The time at the dosing and weighing machines consists of two parts: the actual process time in which dosing and weighing takes place and a removal time in which the machine is unloaded. Figure 2.10 shows three cases of the relation between dosing and unloading (LOS). The total time at a dosing and weighing system is from the start of dosing, until the finish of unloading. This time can be more or less than the sum of the two parts, see Cases 1 and 3 in Figure 2.10. Unloading can only start if the succeeding buffer is available. In Case 1, unloading starts before dosing finishes. Between each two ingredients on a dosing and weighing system, the ingredients can be released to the buffer below, if this buffer is available. An advantage of this intermediate unloading, at dosing and weighing system 2 and 3, is that the succeeding stage, the hammer mill, can start earlier. Recall from Section 2.1 that the ingredients of dosing and weighing system 1 are not processed at the hammer mill.

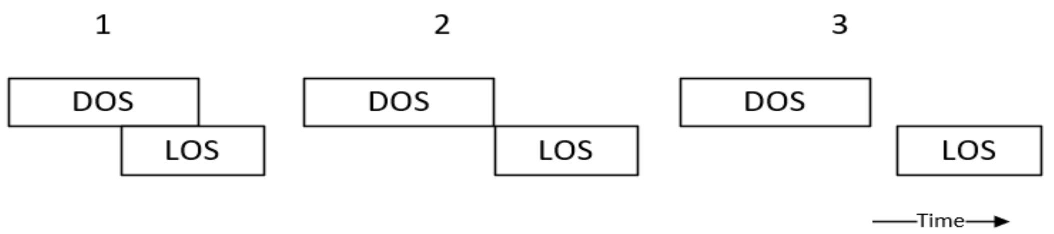

FIGURE 2.10: RELATION DOSING AND UNLOADING, THREE CASES

We determined for each job the relation between dosing and unloading for each dosing and weighing system. For a large fraction of jobs, unloading starts later than dosing stops, because succeeding stages are blocked. This is an indication that the dosing and weighing systems are not likely to be the bottleneck. Table 2.1 shows the fraction of jobs per case per dosing and weighing system.

TABLE 2.1: FRACTION OF JOBS PER CASE PER DOSING AND WEIGHING SYSTEM

Case	DW1	DW2	DW3
1	8.9%	17%	1%
2	15.5%	34.9%	15.5%
3	75.7%	48%	83.5%

In Case 1 of Figure 2.10, there is a minimum time between the finish of dosing and the finish of unloading. This time is independent of quantity or product and takes on average 51 seconds at DW2 (99%-CI on average: [00:00:51– 00:00:51]) and on average 39 seconds at DW3 (99%-CI on average: [00:00:39– 00:00:40]). In Cases 2 and 3, unloading takes on average 53 seconds (99%-CI on average: [00:00:53 – 00:00:53]) for both dosing and weighing system 2 and 3.

The dosing and weighing process time depends on the product and quantity. If more volume of a product is produced, more ingredients need to be dosed. Different products consist of different ingredients. The more ingredients a product has, the more time dosing takes. Each ingredient is taken from a different silo. The availability of ingredients also influences the process time of dosing and weighing system 2 and 3, which are both used to dose ingredients that need to be grinded at the hammer mill. Recall from Section 2.1 that each dosing and weighing system has its own silos. Which ingredient is stored in which silo above which dosing and weighing system changes over time. In the worst case, all ingredients need to be dosed and weighed via one dosing and weighing system. Of course, this takes more time to weigh the ingredients before the hammer mill, than in case the workload is evenly divided over dosing and weighing system 2 and 3. To illustrate the variability in the number of ingredients that needs to be dosed at for example dosing and weighing system 2, Figure 2.11 shows how often a certain number of ingredients is dosed for a production order, for the 5 most produced products. For example, for product 20175, most often 8 ingredients are dosed after each other at dosing and weighing system 2, but also 6 or 12 ingredients are chosen sometimes.

The variability in the number of ingredients per dosing and weighing system, together with the dependency on quantity and product, makes it very complex to determine the process times without modelling the dosing per ingredient in detail. Earlier in this section, we already identified that the dosing and weighing systems are not likely to be a bottleneck. In Section 4.1, we explain more about why we do not take the dosing and weighing systems into account when modelling the process. Because we do not take them into account, we do not determine their process times.

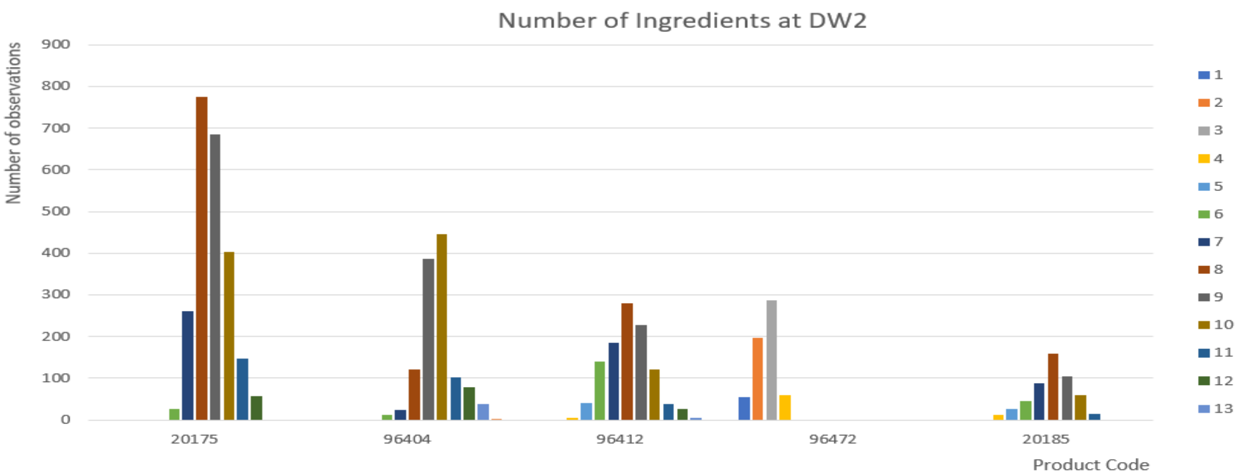

FIGURE 2.11: NUMBER OF INGREDIENTS AT DW2 FOR THE 5 MOST PRODUCED PRODUCTS

Hammer mill

Around the hammer mill there are some stages that are strongly interrelated. First, recall from Section 2.1 that the ingredients are divided over two hammer mills. Both hammer mills are working on the same product simultaneously. Therefore, we speak of one hammer mill unit in this section. The hammer mill is running as long as there are ingredients in buffer 3 or 4. Not all the time, it actually grinds ingredients. This depends on the sieve and succeeding buffer. The hammer mill can only grind ingredients if there are ingredients going through the sieve first. Figure 2.12 shows how the time at the hammer mill relates to the preceding and succeeding processes. The start of the hammer mill is equal to or later than the moment that the first particles enter buffer 3 or 4. However, the actual grinding of ingredients starts after the sieve has started. The sieve can only start when buffer 6 is available. So, after a product finishes at the hammer mill, the sieve must wait until buffer 6 is empty before it can start with the next product. However, the time at the hammer mill in the system for the next product already starts when the previous product finishes. As we said earlier this section, the hammer mill keeps running. Therefore, we decide to define the process time at the hammer mill as the difference between the start time of sieving and the stop time of the hammer mill. Hence, the sieve and hammer mill are aggregated. This is also a modelling decision. Chapter 4 explains more about modelling decisions.

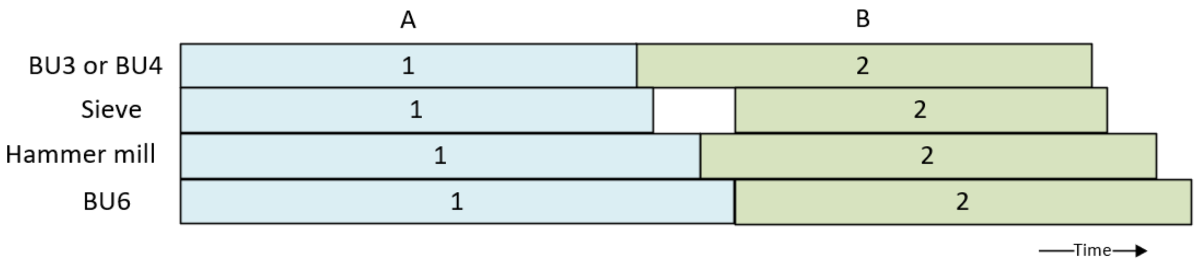

FIGURE 2.12: HAMMER MILL RELATIONS START/STOP; (A) FIRST PRODUCT, (B) SUCCEEDING PRODUCT

Figure 2.12 shows that for the first product all stages of the grind and mix line start at the same time in the MES. It only takes a few seconds for the first ingredients to reach buffer 6. For succeeding products, the start of the different steps depends on the availability of resources. The time between buffer 3 or 4 is empty and the hammer mill finishes is quite constant. After removing outliers, this time is on average 1:42 minutes (99%-CI on average: [00:01:42 – 00:01:42]). The time between the hammer mill finishes and buffer 6 is empty has more variation. Like the dosing and weighing system, buffer 6 can start unloading before or after the hammer mill finishes, depending on whether the mixer is available. Buffer 3 or 4 should also be empty before unloading starts. The time that unloading of buffer 6 takes, depends slightly on the quantity of the job. Therefore, data is grouped per 100 kg and outliers are removed, as explained in the introduction of this section. We fit a regression formula through the median values per group. The formula has a R^2 value of 0.96. Figure 2.13 shows the fit between the regression formula and the real median values.

$$Unload\ time\ buffer\ 6\ (seconds) = 2.55(10^{-3}) * Quantity + 20.13$$

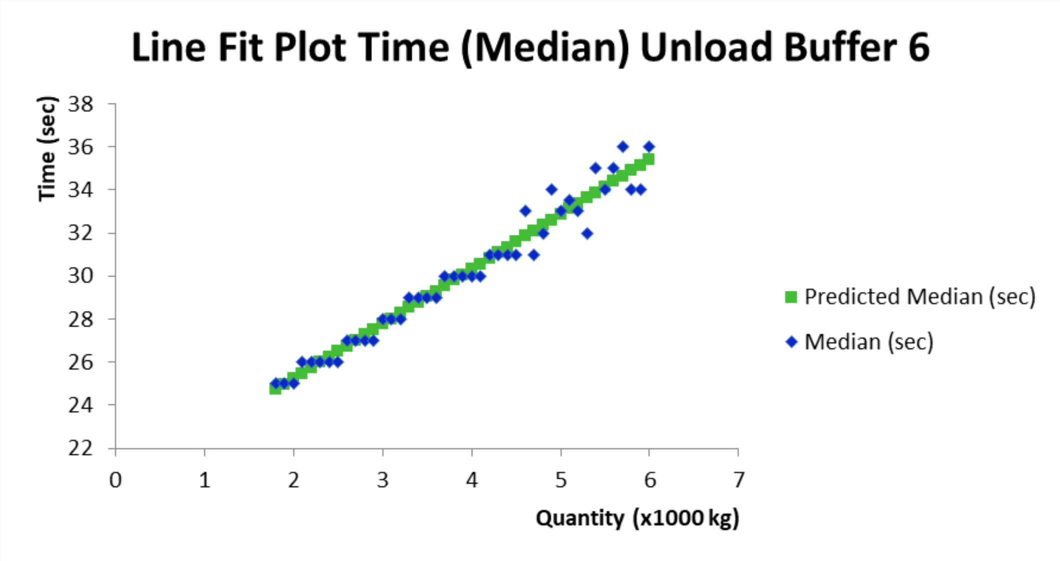

FIGURE 2.13: LINE FIT PLOT UNLOADING BUFFER 6

In Figure 2.13, we see more variation for the higher quantities. This is because for higher quantities, most is produced in batches of exactly 5000 kg or 6000 kg. So in between, we do not have that much data. For lower quantities, there is more variation in the batch sizes. Hence, we have more data per group and a more reliable median value. This median value for the time in seconds is rounded to integer values. Hence, in the figure above we see multiple dots at the same height.

The unload time by the regression formula is the time unloading takes. However, if unloading starts before the hammer mill is finished, it can take more time. In this case, it finishes on average 6 seconds after the hammer mill finishes (99%-CI on average: [00:00:06 – 00:00:06]). Hence, if unloading starts before the hammer mill is finished, its finish time is the maximum of 6 seconds after the finish time of the hammer mill, or the time according to the formula.

The process time on the hammer mill only depends on the quantity to produce. The more to produce, the more time it takes to grind it. The data is again grouped per 100 kg and outliers are removed. A regression formula is fit through the median values per group. The formula has a R^2 value of 0.96. Figure 2.14 shows a line fit plot illustrating the fit between the formula and the real median values.

$$Process\ time\ hammer\ mill\ (seconds) = 2.97(10^{-2}) * Quantity + 153.4$$

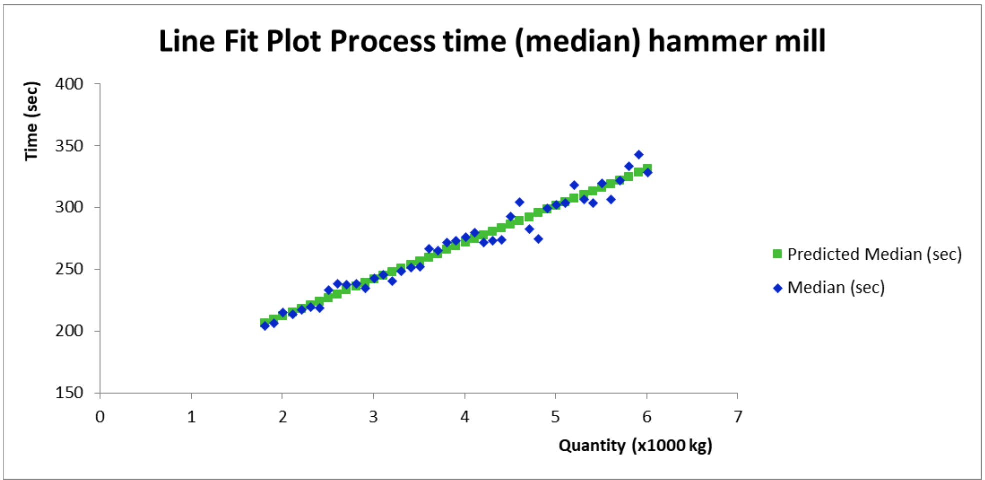

FIGURE 2.14: LINE FIT PLOT PROCESS TIME HAMMER MILL

Mixing 1

The first mixing process, recall Figure 2.2, consists of several sequential procedures. Figure 2.15 shows them. The unloading of buffer 6 starts if the mixer is available and the used buffer 3 or 4 is empty. The time unloading takes depends on whether unloading has started before the hammer mill finishes and on the quantity. After the unloading of buffer 6 finishes, the unloading of WCY1 can start. WCY1 is a unit that blows the minerals from buffer 1 into the mixer. This takes on average 43 seconds (99%-CI on average: [00:00:43 – 00:00:43]). During the loading of the mixer, the mixer already mixes the ingredients. When all ingredients are in the mixer, the mixing program consisting of 3 phases starts. First, a pre-mixing phase is possible. In Deventer, this phase takes 1 second for each product. Next, a so-called spraying phase starts. During this phase, liquids are added. This takes 60 seconds, the speed at which the liquids are added is automatically regulated such that it finishes within 60 seconds. If no liquids must be added, the spraying time is 1 second. The last mixing phase is the post-mixing phase. If no liquids are added, this takes 10 or 60 seconds. When liquids are added, it takes 60 seconds. Between the mixing phases, it takes a few seconds to start the next phase. Table 2.2 shows the average total time for each mixing program. Three different programs can be identified; each program consists of a different combination of the spraying and post-mixing time. The last column shows a 99% confidence interval on the average.

TABLE 2.2: TIMES MIXING 1

Program	Spraying Time	Post-mixing Time	Total time program	Confidence interval
1	00:00:01	00:00:10	00:00:17	[00:00:17 - 00:00:17]
2	00:00:01	00:01:00	00:01:07	[00:01:06 - 00:01:08]
3	00:01:00	00:01:00	00:02:16	[00:02:16 - 00:02:16]

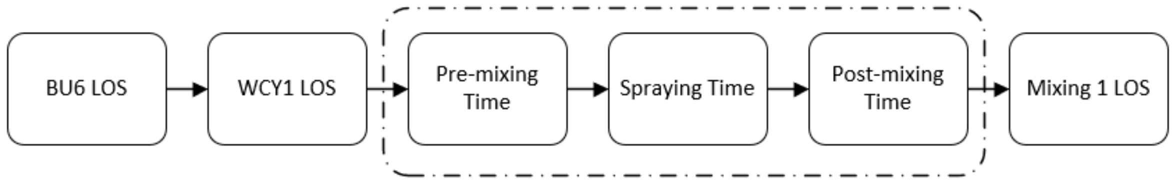

FIGURE 2.15: MIXING 1 PROCESS

Only after the mixing program finishes, unloading can start. Unloading of the mixer takes on average 1:01 minutes (99%-CI on average: [00:01:01 – 00:01:01]). Unloading can only start if the succeeding buffer, buffer 7, is available. If not, the product must wait inside the mixer.

Mixing 2

The last stage of the grind and mix line is the second mixing stage, see Figure 2.2. In the MES, buffer 7 and mixing 2 are the same unit, succeeded by the elevator transport system (see Figure 2.16). Although buffer 7 and the second mixer are the same unit in the MES, the second mixer does not always start at the same time as buffer 7. This depends on the succeeding elevator. After mixing 2 finishes a product, a new product can be loaded in buffer 7. The elevator transports the last particles to their destination. If the next job is a different product, the elevator has a so-called 'nalooptijd' in Dutch. This is a predetermined amount of time it continues to make sure no product is left inside the boxes of the elevator. After this time, the next product can be mixed. On average it takes 1:56 minutes (99%-CI on average: [00:01:56 – 00:01:57]) before the next job can be mixed, in case of a 'nalooptijd'. If the next job is the same product, it only takes 7 seconds (median value) before the next job can start at the mixer.

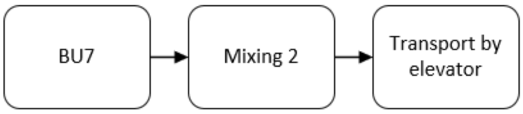

FIGURE 2.16: BU7/MIXING 2

Contrary to the first mixing stage, the second mixing stage is a continuous process; liquids are added while the product flows through it. Thereby, the second mixer regulates how fast buffer 7 can unload. The time that mixing 2 takes, depends on the quantity. The mixer has a certain speed, so more product means more time. The amount of liquids to add does not influence the total mixing time. Based on the speed of the mixer, the system calculates how much time it approximately takes to mix the job. Based on the amount of liquids, the liquid dosing speed is regulated such that the amount of liquids is evenly spread over the product. To determine the processing time, the data is again grouped per 100 kg and outliers are removed. A regression formula is fit through the median values per group. The formula has a R^2 value of 0.98. Figure 2.17 shows the fit between the formula and the real median values.

$$\text{Processing time mixer 2 } (seconds) = 4.35(10^{-2}) * Quantity + 56.29$$

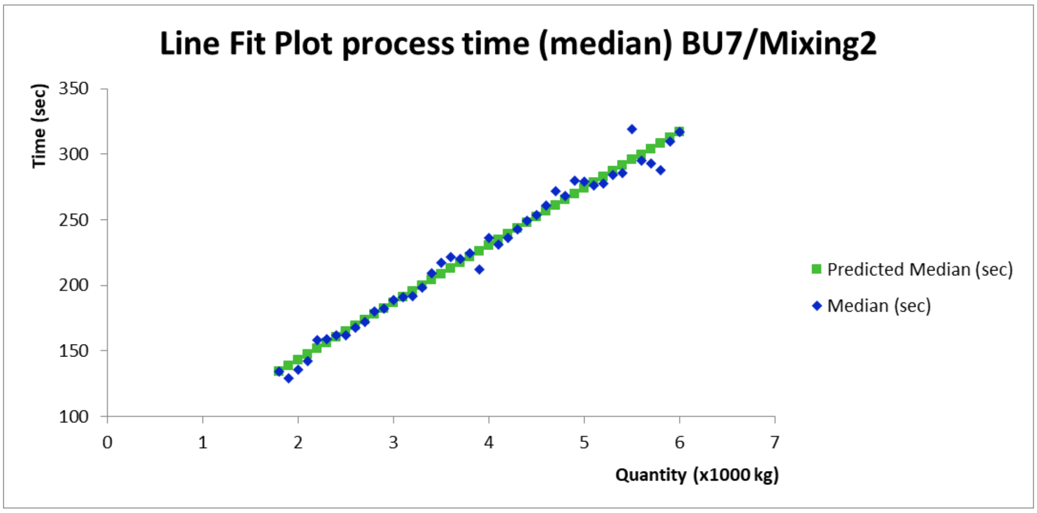

FIGURE 2.17: LINE FIT PLOT PROCESS TIME MIXER 2

Pelletizing

The pelletizing process consists of three stages, which together form a continuous process. Figure 2.18 shows the time relations between the stages. First pre-compaction starts, when the first parts move through the pre-compacter, pelletizing starts. The pellets are pressed and fall into the cooling machine. The cooling machine is a large buffer through which air blows. Pellets need to stay for a while in the cooler. Therefore, a target fill level of 75% is used. Every time the level grows above 75%, some pellets are released, to be around 75% again. Pre-compaction and pelletizing finish shortly after each other; cooling down however, takes some more time to finish. The cooling machine, filled for 75% needs to unload, which takes several minutes. Only when the cooling machine is empty, pre-compaction of the next job can start.

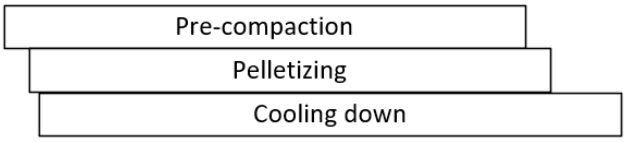

FIGURE 2.18: THE PELLETIZING PROCES IN TIME

The pre-compactor and press process the job at a certain speed, which is called capacity. The capacity is measured in tons per hour. Pellets needs to have a certain hardness and quality. Therefore, settings as the capacity, amount of steam, power (kW), and temperature are important. Different products have a different composition and thereby are processed at a different capacity. This capacity is prescribed in the receipt of the product. However, sometimes circumstances can force the operator to produce at a lower capacity, e.g. ingredients of bad quality. On the other hand, there are operators who try to find the limits by producing as fast as possible, which also can cause an increase in disruptions.

We group the data per line per product and delete outliers. We calculate the actual capacity of a production by dividing the batch size by the duration, after correcting the batch size and duration for the effect of the start-up program (see below). Appendix B shows the average capacity per product per press line in ton per hour.

The pre-compactor cannot start immediately at full speed, because it will most likely jam. Therefore, several start-up programs are defined for each press line. The line starts at a low speed and increases with a certain amount per minute until it reaches the desired speed level. When several jobs of the grind and mix line are combined in one batch, only the first one needs this start-up program. Table 2.3 shows the start-up programs. In total, the software allows 8 different programs per line. However, only three are used in practice. The values are in ton per hour. The 'start' value represents the capacity at which production starts. The 'p/min' value shows the capacity, in number of tons per hour, increase per minute. If we want to produce at 6 ton/h at line 2 with program 1; we start with a speed of 4 ton/h and after 2 minutes we reach the desired speed of 6 ton/h. Program 1 is the fastest program used per line. Program 2 and 3 result in a slower start-up but are sometimes needed.

TABLE 2.3: START-UP PROGRAMS, START VALUES AND INCREMENT PER MINUTE (VALUES IN TON/HR)

Line	1		2		3		4/5	
Start-up Program	Start	p/min	Start	p/min	Start	p/min	Start	p/min
1	4	1	4	1	4	1.5	4	1
2	3	1	3	1	3	1	4	1
3	2.5	0.5	2.5	0.5	3.5	0.5	3	1

The time between the finish of the pre-compactor and the moment the cooling machine is empty, is a kind of removal time. Only after this time the next job can start at the pre-compactor. When multiple jobs of the same product are produced sequentially, the cooling machine does not need to unload intermediately. Table 2.4 shows the average removal time per line and a 95% confidence interval on the average. Line 3 and 5 have a shorter removal time. This is because of the speed and capacity of the elevators per line. Each line has its own elevator. We only take the data where the fill level of the cooling machine is 75%, which holds for 96% of the data.

TABLE 2.4: AVERAGE REMOVAL TIME PER LINE (FILL LEVEL 75%)

Line	Average	Confidence interval
1	00:10:19	[00:10:16 - 00:10:22]
2	00:09:42	[00:09:37 - 00:09:46]
3	00:06:20	[00:06:16 - 00:06:24]
4	00:09:02	[00:08:55 - 00:09:09]
5	00:06:04	[00:05:58 - 00:06:11]

Coating and transport to silo

Every time the cooling machine releases some pellets, they are transported to the upper floor. At the upper floor a sieve separates dust from pellets and a coater coats the pellets before chains transport them to their silo. The capacity of the coater and of the transportation chain is higher than the capacity of the pre-compactor/press. Therefore, their process time depends on the speed of the pelletizing stage. We do not model the coater and transportation chains; we explain this decision in Section 4.1. Hence, we also do not need their processing times.

2.5 PERFORMANCE AND BOTTLENECKS CURRENT SITUATION

To measure the current performance and to identify bottlenecks, we look at the utilization of machines. From the data of Section 2.4, we use four weeks in which no large maintenance operations were done. Table 2.5 shows the weeks and the total production time per week. Figure 2.19 shows the weight produced per week at the grind and mix line and how much is pelletized. The difference between these two is the amount of meal produced, which is only 10% of the total amount. We see quite some differences in the amount produced per week. The produced weight in the fourth week is for example smaller than in the second, despite a larger total processing time.

TABLE 2.5: TIME PER WEEK

Week	Total Time
20-25 Nov	128:32:01
3-9 Dec	136:25:32
21-27 Jan	135:49:22
18-24 Feb	139:40:08

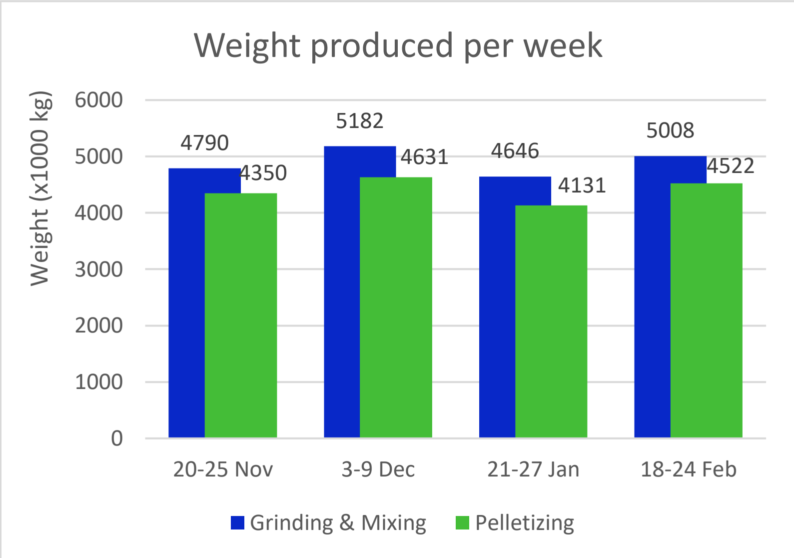

FIGURE 2.19: WEIGHT PRODUCED PER WEEK

Figure 2.20 shows the utilization of the dosing and weighing systems. The blue bars represent the fraction of total time (Table 2.5) that the machine contains a product. The green bars represent the fraction of time that the machine is actually processing or unloading the product. The difference between the blue and green bar is because the product needs to wait until the succeeding stage is available and it can be unloaded. We see a lot of waiting time at the dosing and weighing systems.

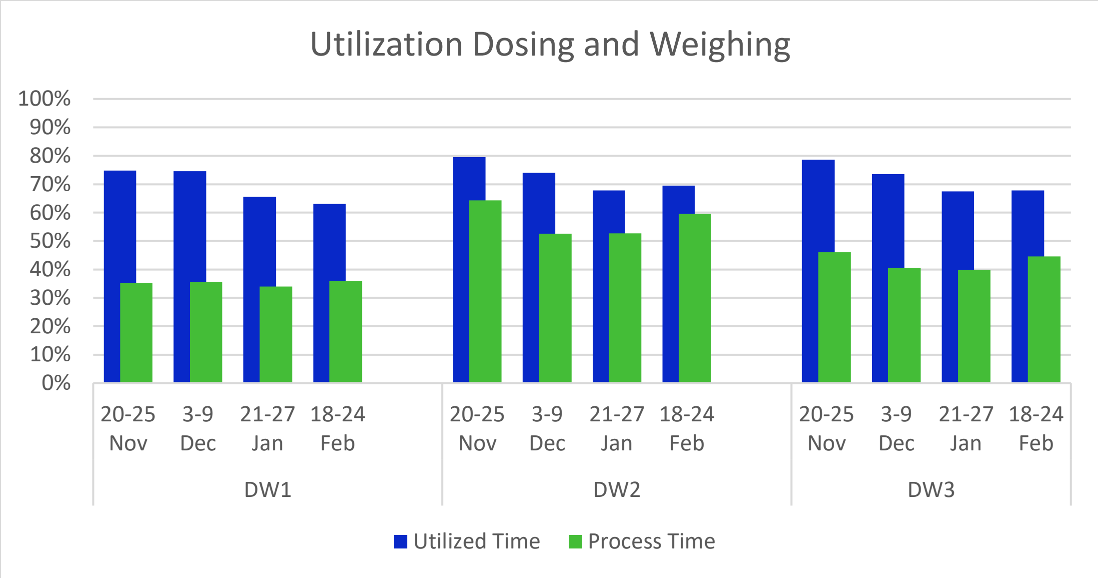

FIGURE 2.20: UTILIZATION DOSING AND WEIGHING SYSTEMS

Figure 2.21 shows the utilization of the other machines of the grind and mix line. The total processing time at the dosing and weighing systems is lower than at the hammer mill or mixers. The difference between the utilized time and the processing time is lower than for the dosing and weighing systems. Hence, the hammer mill and mixers have to wait less.

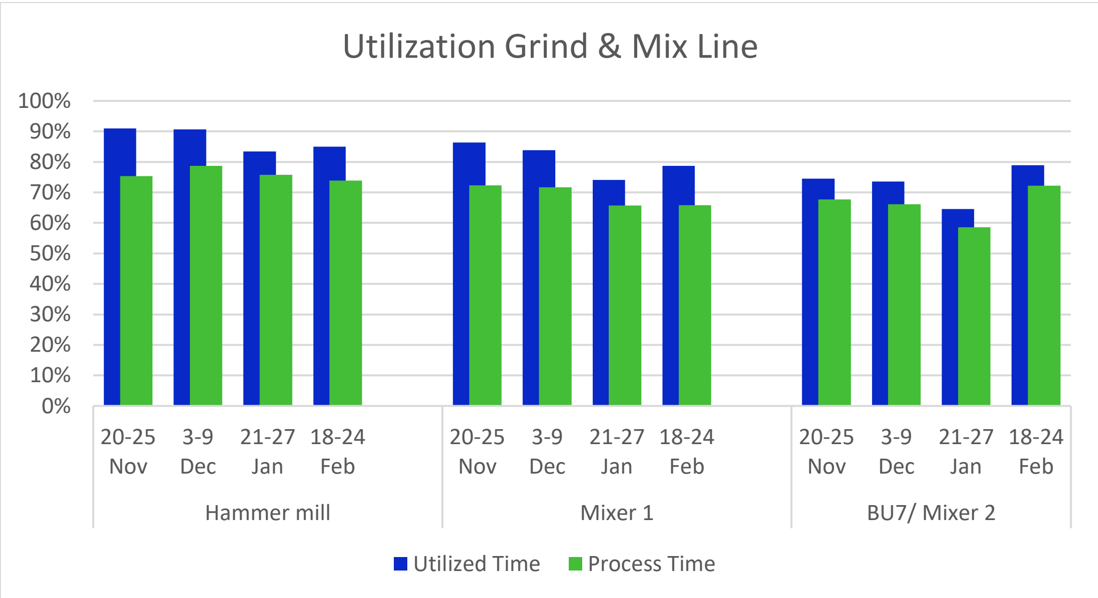

FIGURE 2.21: UTILIZATION GRIND AND MIX LINE

Based on the process time, we conclude that the hammer mill is the bottleneck of the grind and mix line. The waiting times at all machines of the grind and mix line indicate that improvement at the grind and mix line is possible, by optimally sequencing the jobs. Thereby, we reduce the time that an order needs to wait until the next machine is available.

Figure 2.22 shows the utilization of the press lines. The green bars represent the fraction of total time of the week that the press is pelletizing pellets. The blue bars also include the time that it takes to empty the cooling machine. Recall that if the cooling machine is not empty, the next job cannot start at the press.

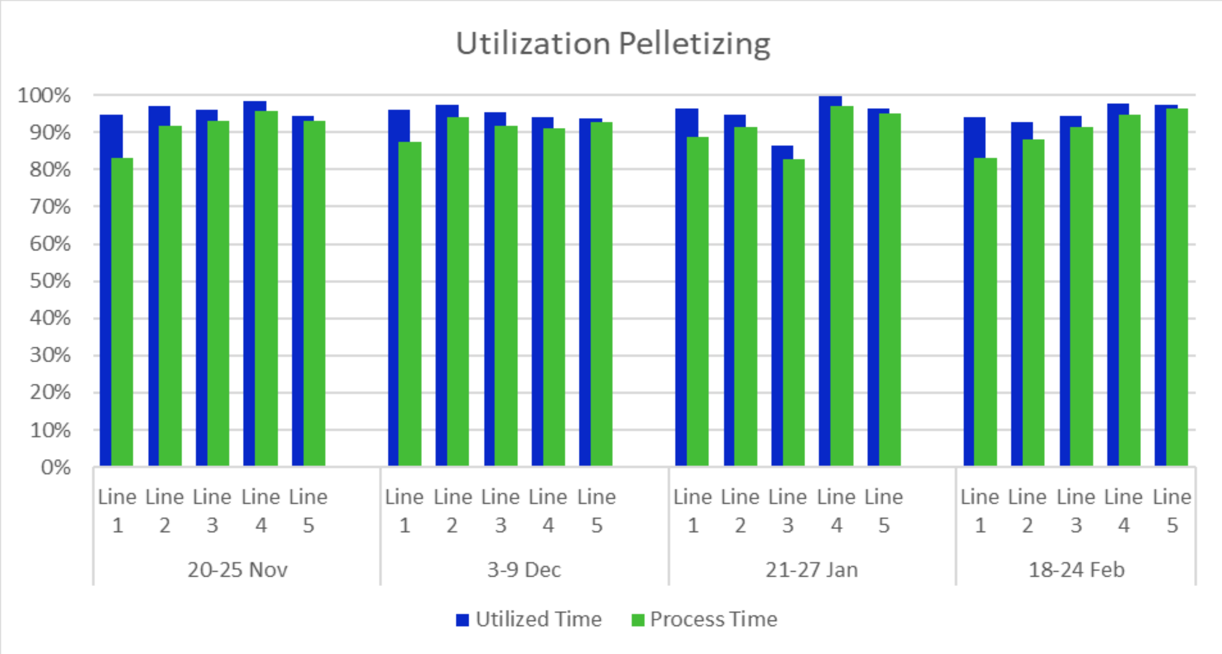

FIGURE 2.22: UTILIZATION PRESS LINES

We see that line 1 has the lowest utilization based on actual process time. The utilized time of line 1 is more or less the same as for the other lines. Hence, the total time it takes to empty the cooling machine is the highest for line 1. In the current production, line 1 pelletizes most make-to-order (MTO) products. In general, these are produced in smaller batches than make-to-stock (MTS) products. For MTS products, more is produced than ordered and the silo is refilled to its maximum amount. For MTO products, only the required quantity is produced. Because of the smaller batches at line 1, more different products are pelletized, and the cooling machine must unload more often. During this time, it cannot pelletize new products, so the utilization decreases. Line 4 and 5 pelletize almost only MTS products in large batches. Therefore, the amount of time to empty the cooling machine is low and the utilization based on actual process time is high. Figure 2.23 shows the total unload times of the cooling machine per line per week.

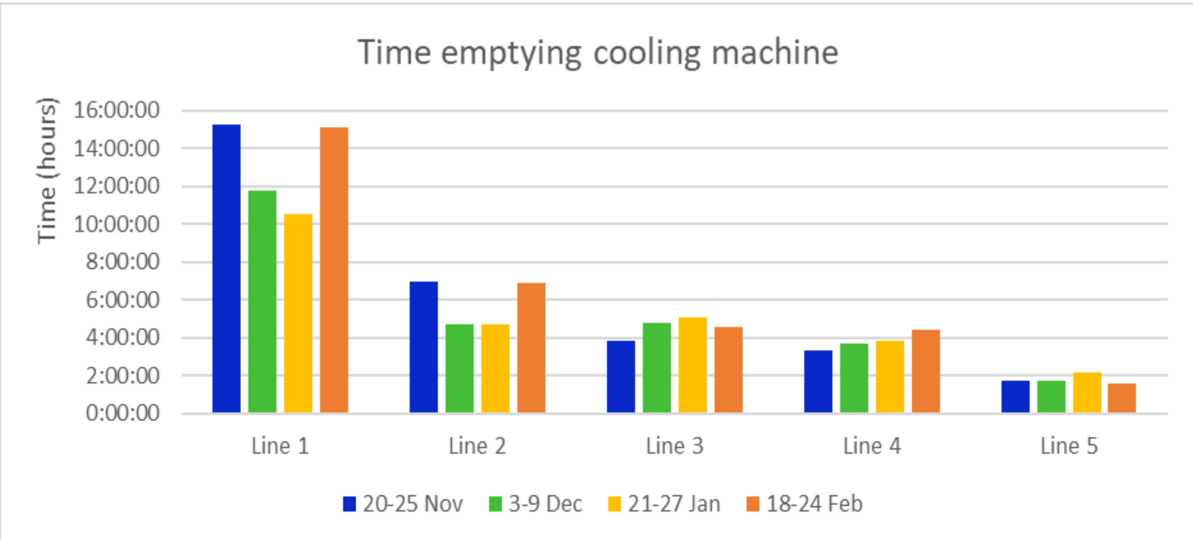

FIGURE 2.23: TOTAL TIME TO UNLOAD THE COOLING MACHINE PER LINE PER WEEK

Figure 2.24 shows the weight produced per week for each press line. Recall from Section 2.1 that the lines with an expander, line 4 and 5, can produce at a higher speed. In Figure 2.24, we see that they indeed produce more than the first three lines. From Figure 2.22 we know that their utilization is about the same, so the higher production is because of a higher speed.

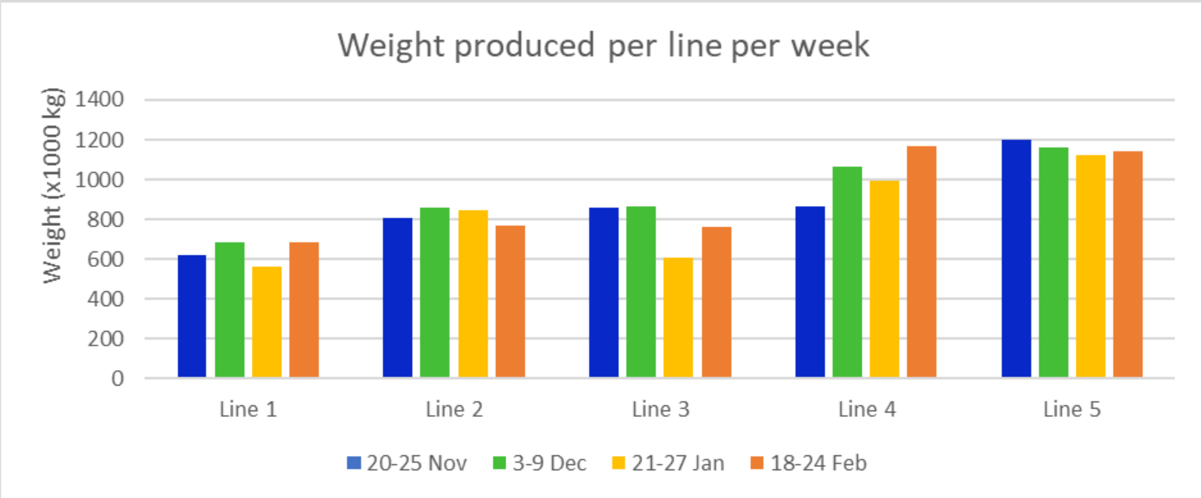

FIGURE 2.24: WEIGHT PRODUCED PER PRESS LINE PER WEEK

When we compare the grind and mix line with the press lines, we conclude that the press lines are the bottleneck for ForFarmers in Deventer. The utilization of the grind and mix line is lower than 80%, based on the actual process time. For the press lines, the utilization is always higher than 80%. Sometimes, even close to 100%.

2.6 Conclusion

This chapter answers the first research question: *How is the production and production planning currently organized and what is the current performance?* It starts with an extensive description of the production process. All products are grinded and mixed, after which meal products are stored and pellet type products are pelletized before they are transported to their silo. Important to note is the limitation in resources for storing a product. There are five chains that can be used, but not each silo can be reached by each chain. Also, some silos cannot be reached simultaneously.

The current production planning is based on intuition and a dynamic collaboration between operators responsible for different parts of the process. The production is partially make-to-order (MTO) and partially make-to-stock (MTS). Meal products are MTO. MTO products are produced short before their due date, after the transport planner planned the transport. MTS products are produced when more is ordered than currently in stock. More is produced than needed and the stock is replenished to its maximum amount. If there is no direct need for a product, an MTS product is chosen which is expected to run out of stock shortly. The order in which products are produced is determined on intuition and experience of the operator. At the grind and mix line, jobs of maximum 6000 kg are processed, which are batched at the press lines.

We identified four important constraints for the production planning. (i) The storage capacity for meal and MTO products is limited. Hence, they can only be produced shortly before their due date. (ii) Not all MTS pellets can be pelletized simultaneously because they need the same chain to reach their silo. (iii) Not each product can be produced at each line because of quality issues and the type of pre-compactor. (iv) Only one buffer is available between most stages of the grind and mix line. If a machine cannot unload, it is blocked.

Besides processing and setup times, we also determined removal times. For the hammer mill, the process time and removal time depend on the quantity to produce. For both, data is grouped in groups of 100 kg and a regression formula is fit through the median values per group. At the first mixer, we start with the removal time of the hammer mill. Next, a constant amount of time of 43 seconds is added if minerals need to be added. Then the actual mixing program is done. There are three different programs with different times, taking 0:17, 1:07 or 2:16 minutes. Finally, unloading of the first mixer takes 1:01 minutes. When unloading starts, the succeeding buffer 7 starts and the second mixer if the succeeding elevator is available. The process time at the second mixer also depends on the quantity and is determined by a regression formula, just as for the hammer mill.

For the press lines, the processing time depends on the quantity and the throughput speed in ton per hour. We have taken the average throughput speed per product per line (see Appendix B). Products do not start immediately at their desired throughput speed. First there is a start-up program where the throughput speed is slowly increased. This period is a kind of setup period. After pelletizing, the pellets need to cool down and when a batch is finished, the cooling machine needs to unload. This time is a removal time and is determined per line. By batching different jobs of the same product, we do only once need a removal time.

In the current situation, the press lines are the bottleneck. The utilization of the press lines is always higher than 80%, sometimes even close to 100%. The utilization of the grind and mix line is below 80%. The lines with an expander, line 4 and 5, produce the highest quantity because they run at a higher speed. Line 1 needs the most time to unload the cooling machine because it is most used for MTO products.

3 LITERATURE REVIEW

This chapter provides an answer to the question: *What is currently known in literature on machine scheduling for flow shops?* Section 3.1 introduces the machine scheduling problem in production and explains how an instance of the problem can be classified. Section 3.2 gives an overview of algorithms from literature, to solve a flow shop machine scheduling problem. Section 3.3 provides some examples of practical applications in literature. Section 3.4 summarizes the most important findings.

3.1 CLASSIFICATION OF MACHINE SCHEDULING PROBLEMS

In production, jobs need to be processed on machines. A job is a product that needs to be processed on one or more machines. The machine scheduling problem in production (MSPP) aims to find a sequence of jobs to be processed on machines in a way that optimizes a set of objective(s) without violating any of the constraints (Graves, 1981). There is a lot of literature available on MSPP and new research is still published today.

To classify the problem at hand, we use the 3-field problem classification $\alpha|\beta|\gamma$ of Graham et al. (1979). The α is used to specify the machine environment. Graham et al. originally split α into α_1 and α_2. The first one, either indicates the type of parallel machines, or the type of shop. $\alpha_1 \in \{\emptyset, P, Q, R\}$, meaning a single machine, identical parallel, uniform parallel, or unrelated parallel machines; or $\alpha_1 \in \{O, F, J\}$, meaning an open shop, flow shop, or a job shop. α_2 shows the number of (parallel) machines.

Vignier et al. (1999) use four alphas to express the machine environment. The first one is used for the type of shop and the second one for the number of stages. The third and fourth are used to express the type of parallel machines and their number for each stage, as $(\alpha_3\alpha_4)^k$. They also introduce the abbreviation FH for the hybrid flow shop (see Section 3.2).

The β is used to specify the constraints and assumptions. Some often-used constraints and assumptions are (Ruiz & Vazquez-Rodríguez, 2010):

- r_j indicate that job j has a release date, so is not available from the start.
- *prmu* indicates that the order of processing is the same in each stage.
- *prec* indicates that there are precedence constraints between operations from different jobs.
- M_j indicates that job j can only be processed on the machines in set M_j at stage k.
- S_{sd} indicates setup times that are dependent on the sequence of operations
- *prmp* indicates that pre-emption is allowed
- *block* indicates that there is limited buffer capacity between stages.
- *recr* indicates that jobs can be processed more than once in one stage.
- *unavail* indicates that machines are not available at all times.
- *no-wait* indicates that jobs are not allowed to wait between two stages.
- $p_j=p$ indicates that all processing times of job j are equal to p.
- $size_{jk}$ indicates that o_{jk} needs to be processed on $size_{jk}$ machines simultaneously.

The γ is used for the objective function. The most common are C_{max}, F_{max}, L_{max}, T_{max}, and E_{max}, respectively the maximum completion time, flow time, lateness, tardiness, and earliness. There are also all kind of total/average (weighted) variants (Ruiz & Vazquez-Rodríguez, 2010).

Based on the discussed terminology, we classify our problem as: FH5, $((1^{(k)})^{3}_{k=1}$, $P10^{(4)}$, $R5^{(5)}|(M_j^{(k)})^{5}_{k=4}$, $S_{sd}^{(3)}$, $S_{sd}^{(5)}$, block, unavail $|L_{max}$.

3.2 Algorithms for machine scheduling in hybrid flow shops

In a hybrid flow shop (HFS), n jobs need to be processed by m stages/workstations. They need to be scheduled such that a certain objective is minimized. The situation at ForFarmers fits the characteristics of an HFS, which are (Ruiz & Vazquez-Rodríguez, 2010):

1. The number of processing stages m is at least 2.
2. Each stage k has $M^{(k)} \geq 1$ machines in parallel and in at least one of the stages $M^{(k)} > 1$.
3. All jobs are processed following the same production flow: stage 1, stage 2, . . ., stage m. A job might skip any number of stages provided it is processed in at least one of them.
4. Each job j requires a processing time p_{jk} in stage k. To the processing of job j in stage k is referred as operation o_{jk}.

In the general HFS, some assumptions are made. These are: (i) all machines and jobs are available at time zero, (ii) each machine can only process one operation at a time and each job can be processed by only one machine at a time, (iii) parallel machines are identical, (iv) pre-emption is not allowed, (v) setup times are negligible and (vi) there is infinite buffer space between stages (Ruiz & Vazquez-Rodríguez, 2010).

Solution approaches to the HFS problem could be classified in three categories: Exact algorithms, heuristics and metaheuristics. Solutions per category are elaborated in respectively Section 3.2.1, 3.2.2 and 3.2.3.

3.2.1 Exact algorithms

To solve the HFS problem to optimality, three exact approaches are often used: Branch and Bound (B&B), Dynamic Programming and Mathematical Integer Programming (MIP). Most research is done in the field of B&B techniques, but often only for small problem instances. Rao (1970) proposes a B&B algorithm for a scenario with two stages. The first stage only consisting of one machine and two identical machines in the second stage. Guirchoun et al. (2005) propose an exact method for this problem, without waiting time allowed between stages. The problem is extended by increasing the number of machines in the second stage to any number M. Lee & Kim (2004) propose a B&B algorithm, which minimizes total tardiness, to solve this problem. Only problem instances with at most 15 jobs could be solved within reasonable time. Gupta et al. (1997) obtain in reasonable time good solutions for a similar problem, with any number M identical machines in stage 1 and only 1 machine in stage 2. Gupta (1988) proves that the hybrid flow shop problem is NP-hard for the two-stage case. For the case with three stages and parallel machines at each stage, Dessouky et al. (1998) propose an extended B&B algorithm. Salvador (1973) adds the no-wait restriction to the general HFS.

To solve the general HFS, Brah & Hunsucker (1991) propose a B&B approach. Their method can solve instances with any number of stages and any number of machines. However, within several hours, only small instances up to eight jobs and three parallel machines at two stages can be solved to optimality. Rajendran & Chaudhuri (1992) choose to only solve the sequencing problem to optimality by a B&B method. At each stage with parallel machines, they use a heuristic approach to assign jobs to the earliest available machine.

All B&B algorithms discussed so far, start their construction of a solution in stage 1, then move to stage 2 and so on. Carlier & NTron (2000) come up with a different approach. At each decision point, the bottleneck and a job are chosen. Trying to find a schedule with a makespan value smaller than the current upper bound.

Some researchers represent the problem as an MIP model and use an IP solver to find a solution. Thereby, implicitly using B&B. An example is Gooding et al. (1994); they use minimizing production costs as objective. Liu & Karimi (2008) report several mathematical models incorporating limited storage, no-wait, batching, unrelated and identical parallel machines, and several optimization criteria.

3.2.2 HEURISTICS

There are three main groups of heuristics that could be used to solve the HFS problem: dispatching rules, divide-and-conquer based heuristics and tailored heuristics. Brah & Wheeler (1998) compared 9 dispatching rules that can be used to find a schedule that minimizes the flowtime or makespan for the m-stage HFS. These dispatching rules are:

- FIFO (First-in, first-out): Select operation that entered queue first.
- LIFO (Last-in, first-out): Select operation that entered queue last.
- SPT (Shortest processing time first): Select operation with shortest processing time.
- LPT (Largest processing time first): Select operation with largest processing time.
- MTWF (Most total work first): Select operation with largest total processing time over all operations.
- LTWF (Least total work first): Select operation with shortest total processing time over all operations.
- MWRF (Most work remaining first): Select operation with largest total processing time for the current stage and next stages.
- LWRF (Least work remaining first): Select operation with shortest total processing time for the current stage and next stages.
- RANDOM: Select operation at random.

The simulation of Brah & Wheeler (1998) shows that the number of jobs, stages and machines per stage have the most influence on the performance. The SPT strategy seems to perform best for the general HFS with makespan or flowtime objective. However, if due dates vary a lot, it is possible that some orders are late.

Lee et al. (2004) analyse several dispatching rules, of which also some lateness objective-based rules, e.g.:

- EDD (Earliest due date): Select operation with earliest due date first.
- SLACK: Select operation with lowest slack, i.e. lowest time between due date and the current time plus the processing times of the remaining processes.
- MDD (Modified due date): Take the maximum of the due date and the current time plus the processing times of the remaining processes, also known as the earliest finish time. Select operation with earliest modified due date first (Baker & Bertrand, 1982).

Several divide-and-conquer heuristics are proposed by researchers. These heuristics divide the problem in smaller subproblems that are solved one by one. Vairaktarakis & Elhafsi (2000) divide the two-stage case into a series of multiple flow shops, thereby reducing the flexibility in routing. Another option is to divide the problem in a single parallel machine problem per stage, as Suresh (1997) did. In his approach, the completion time at stage i, becomes the release time at stage i+1.

A lot of research is done on the shifting bottleneck procedure (SBP) of Adams et al. (1988), an effective divide-and-conquer approach. It gives full priority to the bottleneck stage to maximize its productivity and thereby the productivity of the entire shop (Ruiz & Vazquez-Rodríguez, 2010). For the m-stage problem, Cheng et al. (2001) and Acero-Domfnquez & Patermina-Arboleda (2004) both propose an SBP approach that minimizes makespan. Yang (1998), Lee et al. (2004) and Cheng (2009), all propose an approach to minimize total tardiness. Cheng (2008) also proposes an approach to minimize the number of tardy jobs. Schutten (1998) proposes several extensions for the SBP, incorporating release and due dates, setup times, transportation times, multiple resources, and downtime. Most researchers divide the SBP problem in stages and schedule one stage at the time. However, Phadnis et al. (2003) divide the problem in jobs. After determining the bottleneck, a job is selected, and all its operations are scheduled. This heuristic is called the progressive bottleneck improvement. It performs just as well as the heuristic of Cheng et al. (2001), but is not tested for instances with more than 3 stages.

Other kind of divide-and-conquer heuristics are for example proposed by Guinet et al. (1996). They approach the problem by first defining a sequence of jobs, and next assign jobs at each stage according to this ordering. Santos, Hunsucker, & Deal (1996) propose a similar approach, using regular flow shop heuristics to find job sequences. Thornton & Hunsucker (2004) study the same approach with blocking criteria included. Brah & Loo (1999) also propose a heuristic that separates the sequencing and machine assigning problem, minimizing makespan or flowtime.

Tailored heuristics are also proposed for the HFS problem. However, often for problems not larger than 2 or 3 stages and at least one stage with one single machine, e.g. Gupta (1988) and Gupta & Tunc (1998). Kim et al. (2009) propose heuristics for the two-stage problem with release dates and a product-mix ratio constraint. Compared to the number of heuristics proposed for the 2 or 3 stage problem, of which just a few are mentioned here, there are not that much available for the m-stage problem. Ding & Kittichartphayak (1994) propose several heuristics. Sawik (1993) and (1995) propose tailored heuristics for respectively limited intermediate buffers and no buffers. Botta-Genoulaz (2000) propose a solution for a rather complex problem including lags, setups, removal times and precedence constraints, with a maximum lateness objective. Wittrock (1985) propose the first approach, taking more than one objective into account. He minimizes makespan, taking work in progress (WIP) into account with a kind of lexicographical optimization. Wittrock (1988) adds blocking to his approach.

3.2.3 METAHEURISTICS

The solutions provided by deterministic heuristics, as those in Section 3.2.2, could be improved by so-called metaheuristics. Metaheuristics are general procedures, which provide a solution to many combinatorial optimization problems. In case of HFS, they try to find a permutation of n jobs and build a schedule by assigning jobs onto the machines according to this ordering (Ruiz & Vazquez-Rodríguez, 2010). The most important metaheuristics are simulated annealing (SA), Tabu search (TS), genetic algorithms (GA), ant colony optimization (ACO), artificial immune systems (AIS), and neural networks (NN). The first three types are most frequently used. We elaborate more on them in this section and provide examples of research.

Simulated annealing (SA) is a local search heuristic. A neighbourhood solution is constructed by for example swapping two operations in the current sequence. If this neighbourhood solution is better than the current solution, it becomes the new current solution and a new neighbour solution is chosen. If it does not improve, the neighbour solution becomes the current solution with a certain probability. This probability is very high at the start but decreases over time, decreasing the possibility to accept a worse solution than the current solution. SA tries to escape from local optima. It stops after a predetermined number of iterations, or when no improvement is found after a certain amount of iterations. If in an iteration, a new best solution is found, this solution is saved.

Just as SA, Tabu Search (TS) tries to escape from local optima. TS evaluate all neighbours and takes the best neighbour solution that is not on the Tabu-list. The last k operations are stored on a Tabu-list; solutions on the Tabu-list can in principle not be chosen. However, an aspiration level can be included, e.g. accepting neighbours on the Tabu-list that provide a new best solution. The Tabu-list provides the opportunity to escape a local optimum. The algorithm stops after a predetermined amount of time or when no improvement is found after a predetermined number of iterations.

Voss (1993) uses a dispatching rule as starting point for a TS algorithm. He solves a two-stage problem with a single machine in the second stage and stage-based sequence dependent setup times. Haouari & M'Hallah (1997) propose a SA and a TS algorithm for the two-stage case with identical parallel machines, using simple heuristics for the job assignment at each stage. Wardono & Fathi (2004) use a TS approach and included limited buffer space between stages. Their heuristic outperforms Wittrock's (1988) multi-objective tailored heuristic and Sawik's (1993) tailored heuristic, which both also take limited buffers into account. Finke et al. (2007) simplify the problem by fixing the assignment to the machines for each job. They also use a TS approach that minimizes the sum of earliness and tardiness. Wang & Tang (2009) propose another TS method, incorporating limited buffers and a makespan objective. Naderi et al. (2009) propose a SA approach to solve an HFS with sequence dependent setup times.

Genetic Algorithms (GA) maintain a "population" of solutions and carry out a simultaneous exploration of different parts in the search space (Ruiz & Vazquez-Rodríguez, 2010). Xiao et al. (2000) used GA to solve the m-stage problem by searching through the machine assignment permutation space for a solution with a makespan objective. Ruiz & Maroto (2006) minimized makespan for a m-stage problem with sequence dependent setup times and machine eligibility. His GA algorithm outperformed many heuristics and other metaheuristics. Yaurima et al. (2009) proposed a similar GA, taking limited buffers into account.

3.3 Practical applications

Most of the literature considers theoretical problems. Practical problems are often more complex. They are e.g. larger and consist of more constraints. Some literature incorporates constraints to the problem. However, only just a few of the many constraints possible. In this section, we provide several examples of real world problems from literature and the solution approaches used.

Varadarajan and Sarin (2006) propose various dispatching rules for the semiconductor industry. They propose different (advanced) dispatching rules to sequence the jobs that are waiting for each stage of the wafer fabrication process. Lin and Chen (2015) propose a simulation optimization approach to deal with the complex relations and stochastic nature of the problem. For optimization, they use a GA algorithm and an acceleration technique via an optimal computing budget allocation. By simulation, they perform scenario analyses.

Lin and Liao (2003) solve the problem for a label sticker manufacturing company. They model the problem as a two-stage hybrid flow shop. Stage 1 consist of one machine, stage 2 of multiple parallel machines. They develop their own heuristic, consisting of three steps: (i) determining the production sequence at stage 1, (ii) dispatch the jobs in the queue at stage 2 to one (parallel) machine, and (iii) develop and improve the schedule. For the first step, they first use a dispatch rule to determine a sequence and they improve this sequence by a pairwise comparison procedure that only accepts changes that improve the objective value. For the second step, a FIFO, LPT, or SPT dispatch rule is used. In step 3, they use a procedure based on Tabu search.

Yang et al. (2004) perform a case study at a multilayer ceramic capacitor manufacturer. They use simulation optimization to cover the stochasticity of real world problems and a Tabu search algorithm for optimization.

3.4 Conclusion

This chapter defines the machine scheduling problem in production (MSPP) at ForFarmers as a hybrid flow shop (HFS) problem. Three categories of solution approaches are elaborated: exact approaches, heuristics, and metaheuristics.

Several exact approaches are presented. Most are based on branch and bound (B&B) algorithms and some use a mathematical integer programming (MIP) approach. It may be clear that exact algorithms only solve to optimality for small problem instances, in number of jobs, stages, and parallel machines, within reasonable time. Gupta (1988) proves that the hybrid flow shop problem is NP-hard for the two-stage case. Therefore, no solution can be found within polynomial time, i.e. for larger instances, the running time explodes. Therefore, we conclude that exact methods are not appropriate for this research. The production system at ForFarmers consists of more than three stages and we want to schedule more than just a few jobs.

Three types of heuristics are presented: dispatching rules, divide-and-conquer based heuristics and tailored heuristics. The shortest processing time first is the best performing dispatching rule based on a makespan or flowtime objective but could perform worse for lateness criteria. For optimization on lateness: EDD, SLACK, and MDD are available. Some interesting divide-and-conquer heuristics are presented, of which the shifting bottleneck heuristic is especially interesting to investigate further because a lot of research is done on this heuristic. A lot of tailored heuristics are only for small problem instances. However, some interesting ones are presented that can solve m-stage problems.

In the third category, a lot of Tabu search approaches are presented. From the literature considered, Tabu search seems to be more used than simulated annealing. Some powerful Tabu search algorithms are presented that outperform tailored or divide-and-conquer heuristics. A GA approach seems to be able to even outperform Tabu search and simulated annealing approaches but is also more complex.

4 Modelling the Process

This chapter provides an answer to the question: *How do we find an improved production schedule for ForFarmers?* Section 4.1 explains the problem we want to model and the definition of the stages. Section 4.2 introduces some toy problems, which show the challenges when modelling the process. In Section 4.3, we explain how we create a schedule and in Section 4.4 we explain our optimization strategy. Section 4.5 provides the conclusion of this chapter.

4.1 The problem

To find an improved production schedule, we model the production process. A model of the production process, representing the most important relations between steps of the process, can be used to construct a production schedule. Systematic analysis is possible by comparing different scheduling strategies or the effect of small changes in the schedule on the performance. A production schedule consists of a set of jobs that need to be produced. It represents the assignment to machines and sequence of processing per machine. Recall from Section 2.1 that we have one grind and mix line and 5 press lines. Pellets need to be processed at the grind and mix line and one of the press lines. Meal type products only need to be processed at the grind and mix line. The production lines consist of multiple sequential machines (Figure 2.2).

When a certain amount of a product needs to be produced, it is split up in jobs of at most 6000 kg. This is because the first mixer cannot produce more than 6000 kg at once. Multiple jobs of the same product are combined in a large batch at the press line (Section 2.2). Batching these jobs results in a smaller process time than when these jobs were produced individually (Section 2.4).

A hybrid flow shop machine scheduling model consists of multiple successive stages. The stages can be defined at line or at machine level. We choose to define stages at machine level. Hence, each stage consists of one machine or a group of machines. Recall from Section 2.4 that the processing time per machine depends on the quantity and/or product. Therefore, the machine that has the largest processing time can differ per job, e.g. the hammer mill for one job and the first mixer for another. Recall from Section 2.3 that there is a limited amount of buffer capacity between machines, sometimes even no buffer capacity. Because of the differences in process time and the limited buffer capacity, jobs sometimes need to wait before they can move to the next machine. The amount of time a job must wait depends on the sequence of jobs. To take these relations into account, we need to model the process at machine level.

Recall from Section 3.2 that the production process at ForFarmers fits the characteristics of a hybrid flow shop (HFS) problem. In Section 3.2, we list some assumptions of the general HFS. These assumptions are: (i) all machines and jobs are available at time zero, (ii) each machine can only process one operation at a time and each job can be processed by only one machine at a time, (iii) parallel machines are identical, (iv) pre-emption is not allowed, (v) setup times are negligible and (vi) there is infinite buffer space between stages (Ruiz & Vazquez-Rodríguez, 2010).

The assumptions do not hold in case of ForFarmers. Therefore, we must find a way to deal with them. Below, we list why assumptions do not hold:

i. The plant in Deventer runs for 136 hours per week. Therefore, it is highly unlikely that all machines are available at the time that we start our schedule, because new jobs need to be scheduled after jobs already in production.
ii. A job can be processed by multiple machines at the same time. For example, the press can start before the second mixer finishes (see toy problem 1, Section 4.2).
iii. The parallel machines of the press lines are not identical. There are multiple differences, as discussed in Section 2.1.
iv. Pre-emption is possible at the press lines. Batches are sometimes interrupted to produce another product first. However, we assume that pre-emption is not allowed in our algorithm.
v. At the press lines, changeover times of changing a mould cannot be neglected. Also, the start-up programs take a certain amount of time (recall Section 2.4).
vi. There is limited or no buffer space between stages.

Recall from Section 2.1 that the production process is automated by a Manufacturing Execution System (MES). The MES models the process, described in Figure 2.2, in units. Each unit is a stage of the process that can possess its own job. A unit can be a machine, buffer or an elevator transporting system. Figure 4.1 and Figure 4.2 show the units of the MES. These units are the basis behind our model. We can define each unit to be a stage in our model. However, the model would become rather large and complex because of all relations between units. Therefore, we exclude some units for which our analysis shows that they are never delaying the process. We explain more about this decision and analysis later this section. We also aggregate some units into one stage for the model. We indicate the stages of the model with dashed rectangles in Figure 4.1 and Figure 4.2. Figure 4.1 shows the units of the grind and mix line, Figure 4.2 the units of the succeeding press lines. Reason to aggregate units is that these units always possess the same job, i.e. the first unit waits until the last unit finishes before it starts with the next job. Hence, together they form one sub process, which we model as one stage. We aggregate units around the hammer mill, buffer 7, and at the press lines into one stage.

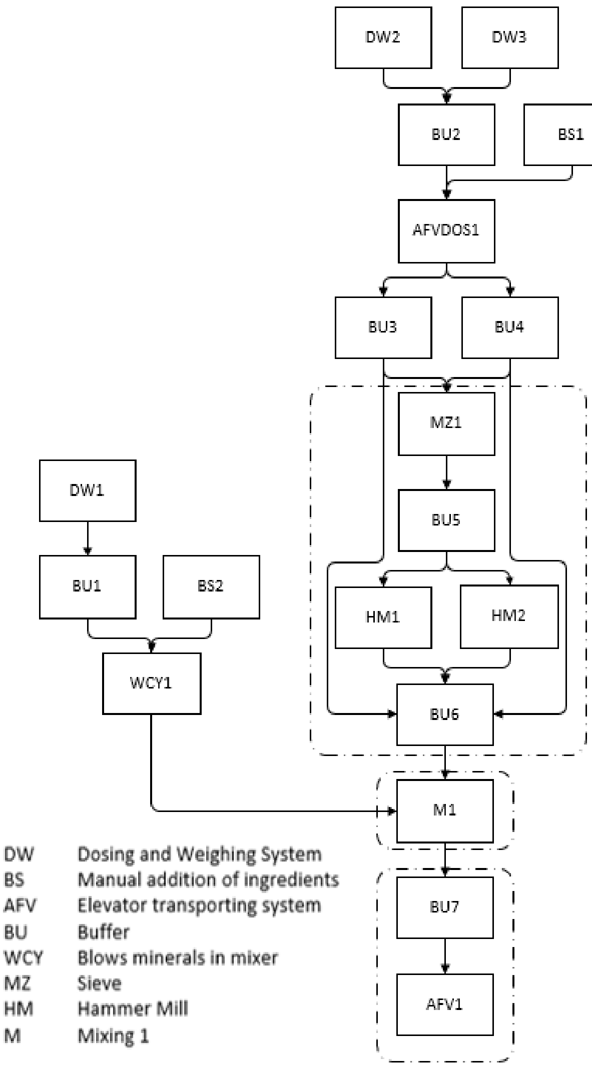

FIGURE 4.1: UNITS MES GRIND AND MIX LINE

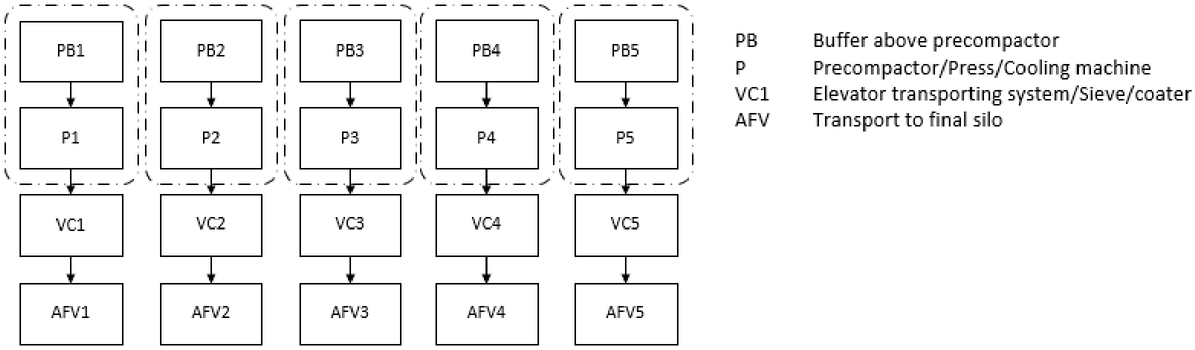

FIGURE 4.2: UNITS MES PRESS LINES

We model the process in five stages. The dashed rectangles in Figure 4.1 represent three stages. The dashed rectangles in Figure 4.2 represent five parallel machines that form one stage. We add a fifth stage, the press silo, to them. We use this stage to connect the grind and mix line to the press lines, to model the intermediate storage, and to cover the limited capacity of the press silos. Toy problem 2 aims to make clear why the press silos are necessary to include. Figure 4.3 shows the five stages and the number of machines per stage.

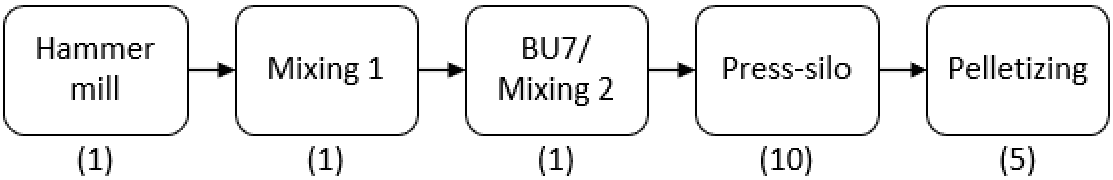

FIGURE 4.3: FIVE STAGES OF THE MODEL AND THE NUMBER OF MACHINES PER STAGE

We exclude the dosing and weighing systems, manual addition of ingredients, coating machines, and transport to the final silo from our model. The number of products where ingredients need to be added manually is limited. We do not model this, and we assume there is always an operator available to do this, without delaying the process. We explain the decision to exclude the dosing and weighing systems, the coating machines, and the transport to the final silo in more detail below.

Excluding dosing and weighing systems

First, we explain why we do not take the dosing and weighing systems into account. The main reason is that the dosing and weighing systems never delay the succeeding steps of the process. To support this claim, we present four arguments:

1. Table 2.1, shows that for 48% to 83.5% of the jobs, there is some waiting time between the finish of dosing and the start of unloading. Figure 2.20 shows that the total process time of the dosing and weighing systems is lower than for the other stages of the grind and mix line, shown in Figure 2.21. Figure 2.20 also shows that there is a lot of waiting time at the dosing and weighing systems, which corresponds to our findings in Table 2.1. Waiting time indicates that a succeeding stage needs more time to process the product. So, the next stage does not have to wait before it can start processing the next job.
2. When we look at the data of our dataset of Section 2.4, we see that for 98% of the jobs at dosing and weighing system 1, the processing time is smaller than at the hammer mill. For dosing and weighing system 2 and 3, 94.6% of the jobs have a larger time at the hammer mill than at both dosing and weighing systems. The remaining 5.4% of the jobs has 1 dosing and

weighing system that takes more time. We found no general explanation for the remaining 5.4% of the jobs to explain why they take more time. Recall from Section 2.5 that the hammer mill is the most time-consuming machine of the grind and mix line. Now we know dosing and weighing takes less time than grinding at the hammer mill, we assume that succeeding stages never have to wait for ingredients to be dosed.

3. The difference between the hammer mill time and the maximum dosing and weighing time of system 2 and 3 also supports the claim that succeeding stages never have to wait on the dosing and weighing systems. On average, this difference is always positive. Hence, the time at the hammer mill is larger. When we create a 95% confidence interval on the average difference per product, we can be for 95% sure for 89% of the products that the difference is positive. For most of the remaining 11% of our products, we only have 2 to 5 data points which results in a wider confidence interval and/or a relatively high standard deviation.
4. By not modelling the dosing and weighing systems, we omit multiple buffers between the dosing and weighing systems and succeeding machines. This reduces complexity because we do not have to think about when these buffers are available or occupied. Otherwise, this would be required to determine when the dosing and weighing system can start with the next job.

Excluding coater and transport to final silo

As Figure 4.2 shows, the last two units of the press line are the coating machine and the transport to the final silo. The coating machine and transport chain (Section 2.1) can process the product at a higher speed than the press. When we compare the processing time at the press with the processing time at the coating machine or the processing time of transport to the silo, we see that for more than 99% of the jobs the processing time at the press is larger. Hence, we conclude that the press takes more time and we can exclude the coater and transport to the final silo. The press never has to stop processing because the succeeding machines did not finish the previous job yet.

4.2 TOY PROBLEMS

We have to deal with some extra complexities, compared to the standard hybrid flow shop (HFS) problem (recall Section 3.2). To explain and illustrate these issues, which make it complex to model the process and to create a schedule, we use some toy problems. In these toy problems, we only create a schedule for a few jobs. We also limit the amount of press lines we use. The processing times at the first three stages, which are of the grind and mix line, are according to the estimations we determined in Section 2.4. For stage 5, we halve the estimated process time. Thereby, the differences between the grind and mix line and the press line become smaller, which increases readability of the schedule and the relations between stages. The first toy problem illustrates the relations between stages. When do they start or finish, and what are the unloading relations? The second toy problem shows the relations in the batching of jobs for the press lines.

Toy problem 1

In the first toy problem we only use 2 press lines instead of 5, each having 2 press silos. We create a schedule for 6 jobs. Table 4.1 shows the jobs with some information. They are all a different product. Four are pellets, two are meal types. The colour corresponds to the colour in Figure 4.4. Figure 4.4 shows a schedule for toy problem 1, scheduling the jobs in the order of Table 4.1.

TABLE 4.1: JOBS TOY PROBLEM 1

Job	Colour	Product Code	Type	Quantity (kg)
1	Orange	20175	Pellet	6000
2	Green	96404	Pellet	6000
3	Light Blue	20216	Pellet	6000
4	Dark Blue	11400	Meal	6000
5	Red	20185	Pellet	6000
6	Yellow	96470	Meal	6000

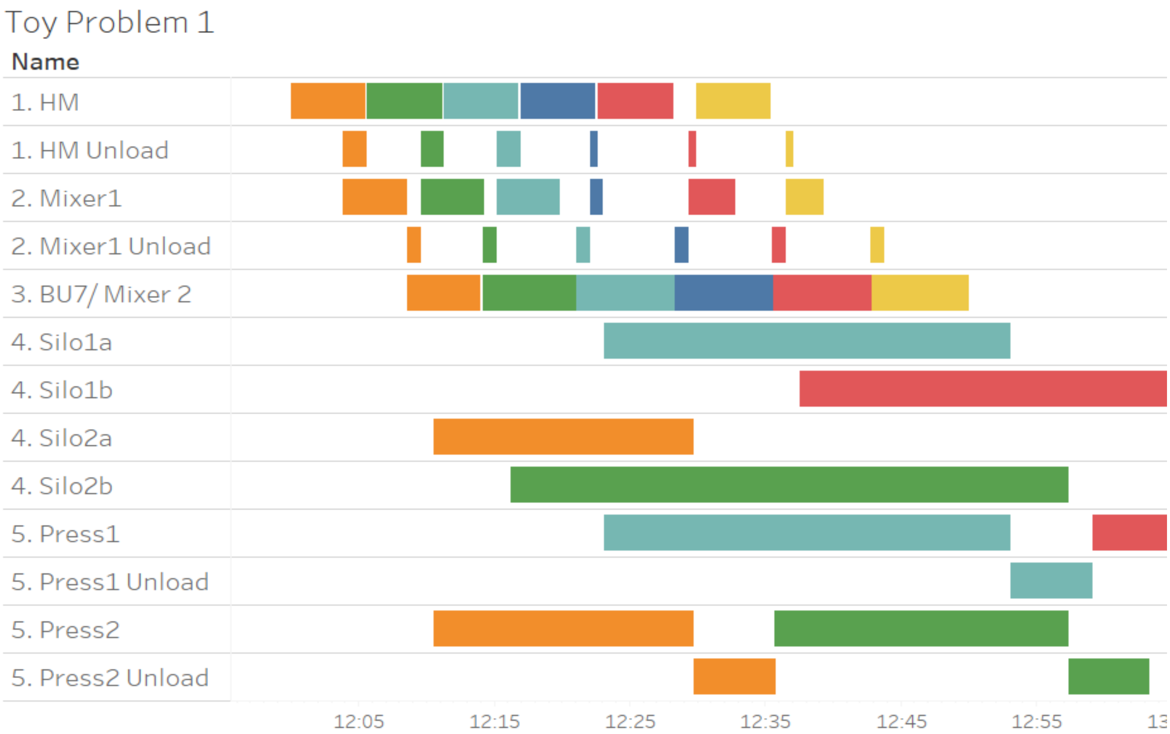

FIGURE 4.4: SCHEDULE TOY PROBLEM 1

Recall from Section 2.4 that stages can have a removal time. The machine needs to be emptied, which takes time. The unloading process can start before the job finishes processing, at the time it finishes processing, or at a later moment in time. Recall Figure 2., showing this relation for the dosing and weighing systems. The third stage does not have a separate unload time, while processing the jobs it unloads and the product is transported to the press silo. Press silos also do not have a specific unload time. The press silo feeds the succeeding press and is empty when the press finishes.

We list the relations illustrated by Figure 4.4 below:

- Stage 1, 2, and 3 cannot start a new job before unloading of the current job finishes.
- Unloading of stage 1, the hammer mill stage, starts a certain amount of time before the hammer mill finishes, if the first mixer is available. Unloading only can start when the preceding buffer 3 or 4 is empty (Section 2.1), which is a certain amount of time before the hammer mill finishes. Unloading takes some time, see Section 2.4. It cannot finish before the hammer mill itself finishes but finishes at least a few seconds later. For the first four jobs in Figure 4.4, unloading starts before processing finishes. For the last two jobs, unloading happens at a later moment in time, because the first mixer is not available immediately. The first mixer is not available immediately because it must wait for the buffer 7 to be available, before it can start to unload.

- When the hammer mill stage unloads, the first mixer stage starts. Recall Section 2.4 for the steps at this stage. Three mix programs are possible, recall Table 2.2. The pellet jobs all have program 3. We see in Figure 4.4 that they have the largest processing time. Job 4 requires program 1, a very short program. Job 6 requires program 2.
- Unloading of the first mixer can only start when the mixing program finishes.
- The second mixer can start immediately when unloading of the first mixer starts. However, the succeeding elevator needs to be available to start processing the job. This elevator has a so-called 'nalooptijd' if the job is of a different product than the previous one (recall Section 2.4). Recall from Section 2.4 that the processing time depends on the quantity. We see in Figure 4.4 that the first job has a shorter processing time, even though all jobs are of the same quantity. This is because it does not have to wait for the elevator.
- A certain amount of time after the first kilograms of the jobs went through the second mixer, they reach the press silo via the elevator transporting system. The press starts at this moment, if it is available. Otherwise, it is pelletizing a job from the second press silo. The job then starts when the press finishes and after the cooling machine is empty. Job 1 and 2 at the second press line in Figure 4.4 illustrate this relation. Recall Section 2.1 and Section 2.4 for a more in-depth explanation of the processes of the press line.
- When the press finishes, the cooling machine needs to unload. But only if the next job is of a different product. Figure 4.4 illustrates this with the 'press unload' process. After unloading finishes, the next job starts to be pelletized.

Toy problem 2

In the second toy problem we only use one press line with one dedicated press silo. We give this press silo a capacity of 12,000 kg. We create a schedule for 4 jobs, all of the same product. Table 4.2 shows these jobs and again some information. Figure 4.5 shows a schedule for the jobs of Table 4.2.

TABLE 4.2: JOBS TOY PROBLEM 2

Job	Product Code	Type	Quantity (kg)
1	20175	Pellet	6000
2	20175	Pellet	6000
3	20175	Pellet	6000
4	20175	Pellet	6000

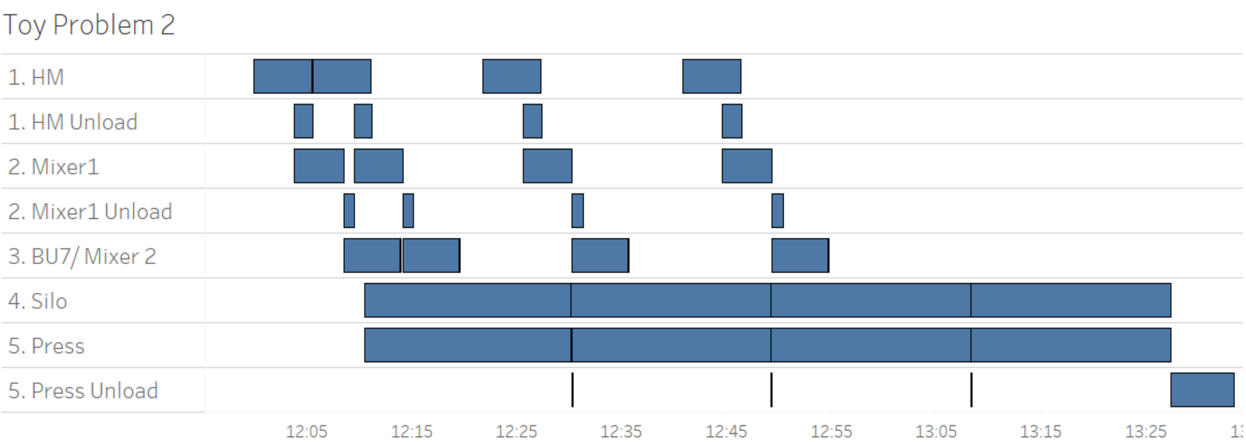

FIGURE 4.5: SCHEDULE TOY PROBLEM 2

With a press silo capacity of 12,000 kg, we can store 2 times a job of 6,000 kg at once. The first job can be stored and starts to be pelletized as soon as possible. The second job also can be stored and is pelletized after the first one, without interruption. For the third job, we have a 'problem'. Already a part of the first job is pelletized, so there is some space available in the press silo. However, we don't know exactly how much. Nevertheless, we know that when the first job finishes at the press, there is enough space in the press silo for another job of 6,000 kg. Now, there are two options at the grind and mix line. (i) We start immediately with job 3 after job 2 and we block stage 3 because the product cannot move immediately to the press silo or (ii) we start job 3 at a later moment in time such that it finishes just when the press silo becomes available. Figure 4.5 illustrates the second option.

The first option is often not an option in reality. Blocking a machine stops the entire line. Thereby, there are more jobs that can be produced in that period. So, if one job would block the line, it is produced at a later moment in time and first some other jobs are produced. For creating a schedule, it is important to determine this moment in time, so that we do not block lines. For job 4 in this toy problem, the same holds as for job 3.

In Figure 4.5, we see that only the last job of a product has an unloading time in stage 5. In case of two succeeding jobs of the same product, there is no unloading of the cooling machine needed.

4.3 Constructing a feasible schedule

In Section 4.1, the stages of our model are defined, and we show that the general assumptions of the HFS problem do not hold. Section 4.2 introduced multiple relations that are important to consider when creating a schedule. In this section, we explain how we create a schedule, given the (complex) relations we identified. Figure 4.6 shows our scheduling algorithm. We first schedule the press lines, followed by the grind and mix line. We explain our decision to schedule the press lines first in Section 4.3.1 and elaborate on our approach to construct a schedule in Section 4.3.2. When scheduling the press lines, we do not take care that the grind and mix line is able to produce the jobs in time. Therefore, the schedule may be infeasible. In Section 4.3.3, we explain why we allow a schedule to be infeasible at first and how we change it to a feasible schedule. After we explained how we create a basic schedule, we extend our algorithm by adding optimization strategies in Section 4.4.

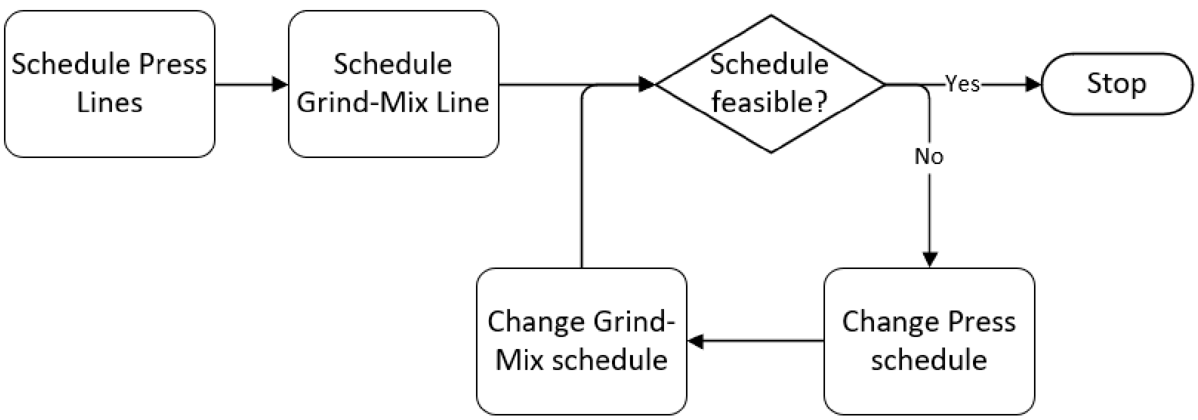

FIGURE 4.6: STEPS TO GET A FEASIBLE SCHEDULE

4.3.1 Scheduling press lines first

The main reason to schedule the press lines first is the limited capacity of the press silos, which are the buffer capacity between the grind and mix line and the press lines. By scheduling the press lines first, we determine when a job can be processed at the grind and mix line, such that it fits in the press silo. A second reason is that we want at least the jobs of the same product, with the same due date, next to each other at the same press lines. This is easier to accomplish when we schedule the press lines first.

The most challenging part in the connection between the grind and mix line and the press lines is when to release a job for the grind and mix line such that it fits in the press silo, without blocking the grind and mix line because it must wait. We choose to prevent blocking when creating a schedule because in general, the operators do not start production at the grind and mix line if they know that the product cannot move to the destination press silo. If the line is blocked, operators cannot react on last minute changes, e.g. producing another product first. Therefore, they delay the start of production until the press silo is available. By scheduling the press lines first, we are able to determine when there is space in the press silo to store a job. Hence, when the job can be produced best at the grind and mix line. Therefore, the job gets a release date and a due date for the grind and mix line. Given the capacity of the press silo and the finish times at the press, we determine when a job may finish at the grind and mix line. Given the minimum process time at the grind and mix line for the job, we determine its release date. We determine the due date for the grind and mix line based on the start date of the press.

Important to note is the assumption that only one press silo can be used for a product. In reality, both dedicated press silos can be used to store jobs of a certain product. We choose to use this assumption because it is complex to implement a rule that uses both press silos, within the limited time for this research. When we would use two silos, we must take care of which silo is currently used by the press and of rules that take care of the question: do I use the second silo, or do I postpone production at the grind and mix line? In the end, it does not change the schedule at the press lines. Only the interval in which jobs can be scheduled at the grind and mix line becomes smaller by our approach for part of our jobs, which can result in a somewhat worse solution. However, the grind and mix line has excess capacity. So, the impact is limited.

If we would start scheduling at the grind and mix line, scheduling the press lines becomes rather complex. First of all, we would also schedule job-by-job, but now we schedule all stages at once for each job, instead of dividing the problem in two sub problems. At stage four we need to select a press silo dedicated to a line where the job is allowed to be scheduled. But if no press silo is available, we block the grind and mix line until an allowed press silo becomes available. If we schedule a job at the press silo such that it is full afterwards, we have to update the release dates of the remaining jobs of the product. These additional procedures take more running time. Getting the jobs of the same product and with the same due date after each other on the same press line also can become more difficult. Whether this is still possible depends on the sequence at the grind and mix line.

The last disadvantage of starting to schedule at the grind and mix line is an important reason for us to start at the press lines when we create a schedule. We want to produce jobs of the same product, with the same due date, after each other at the same press line. An important reason for this is the fact that the press lines are the bottleneck for ForFarmers in Deventer (Section 2.5). Recall, e.g. from toy problem 2, that when jobs of the same product are scheduled after each other, we save start-up time and time it takes to empty the cooling machine. Thereby, we have more time available to pelletize jobs. By creating first a schedule for the press lines, we can relatively simple schedule jobs of the same product at the same press line, without needing all kind of difficult and complex control rules.

4.3.2 Schedule job at a machine

Figure 4.7 shows a logic flowchart that illustrates the steps to create a schedule. The blue rectangles are steps that consist of sub steps. We discuss these sub steps later this section. First, we sort the jobs on their due date. Each job can have a different due date, so we reduce the probability of finishing a job late by producing the earliest due dates first. Recall the EDD-rule in Section 3.2.2. After scheduling the jobs at the press lines, we sort the jobs again on due date for the grind and mix line. The pellet type jobs get a due date for the grind and mix line equal to their start time at the press. For meal products, we use their production due date. Given the release and due date for the grind and mix line, we schedule each job at each machine of the grind and mix line. We schedule the jobs job-by-job at each line. For example at the grind and mix line this is necessary. To determine when a job can start unloading at a certain stage, we need to know the time at which the previous job finishes unloading at the next stage. Therefore, we need to schedule the previous job at all stages of the line first.

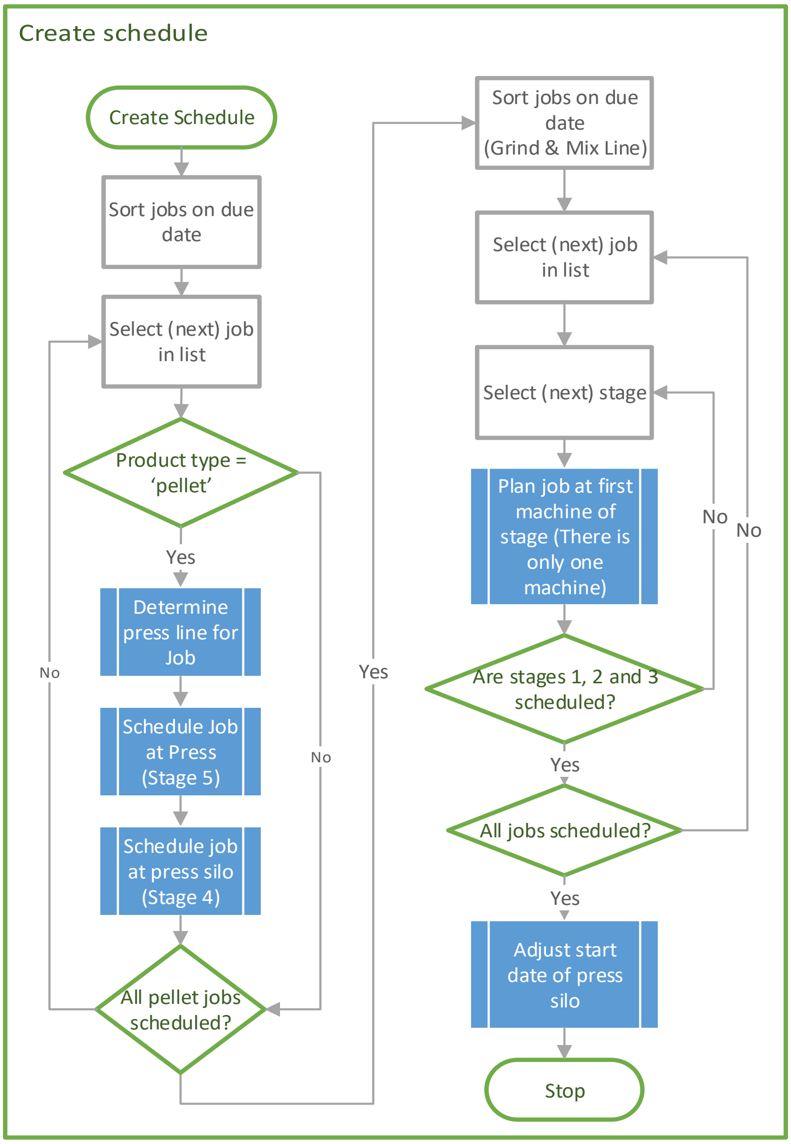

FIGURE 4.7: LOGIC FLOWCHART CREATION OF SCHEDULE

When scheduling jobs at the press lines, we first assign the job to a press line. If the job is of the same product as the previous job, we schedule it at the same press line. Thereby, jobs of the same product are scheduled after each other, in so far they have the same due date. If the jobs are not of the same product, we select the earliest available line out of the lines at which the product can be produced. When multiple lines are available at the same time, we select the line with the smallest production time for the job. In the remaining part of this section, we first explain how we schedule a job at a machine of stage 1, 2, 3, or 5. Next we explain how we schedule the press silos.

Schedule job at machine

After assigning the job to a press line, we schedule the job at the press (stage 5). This is almost the same procedure as scheduling the job at the machines of stage 1, 2, and 3. Before we explain the scheduling of a job at a machine, we first introduce some notation. The purpose of some parameters becomes clearer in the explanation of how to schedule a job at a machine.

i	stage index,	$i = 1, 2, 3, \dots, 5$
j, k	job index	$j = 1, 2, 3, \dots, J$
m	machine index	$m = 1, 2, 3, \dots, M$

Parameters

p_{ijm}	Processing time of job j at machine m in stage i
sd_{ijkm}	Machine based setup time at machine m in stage i, when job k is processed after job j
u_{ijm}	Unload/removal time of job j at machine m in stage i
d_j	Due date of job j
rs_{im}	Minimum amount of time before unloading (removal) can start before finish processing of job at machine m in stage i
rf_{im}	Minimum amount of time before unloading (removal) can finish after job is finished processing at machine m in stage
ts_i	Time after which stage i can start after stage $i-1$ started

Variables

R_{ij}	Release date of job j in stage i
D_{3j}	Due date of job j in stage *3*/ Due date for grind and mix line
S_{ijm}	Start date of job j at machine m in stage i
F_{ijm}	Finish date of job j at machine m in stage i
US_{ijm}	Start date unloading of job j at machine m in stage i
UF_{ijm}	Finish date unloading of job j at machine m in stage i
A_{im}	Time at which machine m in stage i becomes available

Figure 4.8 shows a logic flowchart that illustrates the steps to schedule a job at a machine. When we schedule a job at a machine, we adjust multiple variables, which we use to store the schedule. These variables are defined above. First, we assign the job to the machine, such that we know which job is processed at which machine. Next, we adjust the necessary variables, such as the start and finish dates. We explain per variable how this is done. Recall toy problem 1 showing all kind of relations that determine when a job starts or finishes. We elaborate on these relations in the explanation below.

Start time

The start time at the stages of the grind and mix line, stage 1 to 3, is equal to the maximum between the release date (R_{ij}) and the time the machine becomes available (A_{im}). When the machine is available, and the job is released for its stage, the job can start. Otherwise, the job starts at its release date.

For stage 5 the same rule holds as for the grind and mix line, if the job to schedule is of another product than the preceding one. If the job is of the same product as the preceding one, the start date is equal to the maximum between the release date (R_{ij}) and the finish date (F_{ijm}) of the preceding job at the same line. Thereby, we skip the unloading time. Recall from toy problem 1 and 2 that we only unload the cooling machine for the last job of a product when the jobs of the same product are produced after each other.

Stop time

The finish time at the first stage is equal to the start time (S_{ijm}) plus the processing time (p_{ijm}). At the second stage, we take the finish time of unloading of the previous stage ($UF_{i-1,jm}$) and add the processing time (p_{ijm}) to it. Recall from toy problem 2 that the second stage starts when unloading of stage 1 starts. However, the unloading of stage 1 takes a variable amount of time, depending on whether unloading of stage 1 starts before processing of stage 1 finishes or at a later moment in time. The mixer of the second stage only can start processing when unloading of the first stage finishes. Hence, it makes sense to use this time and to add the processing time to it.

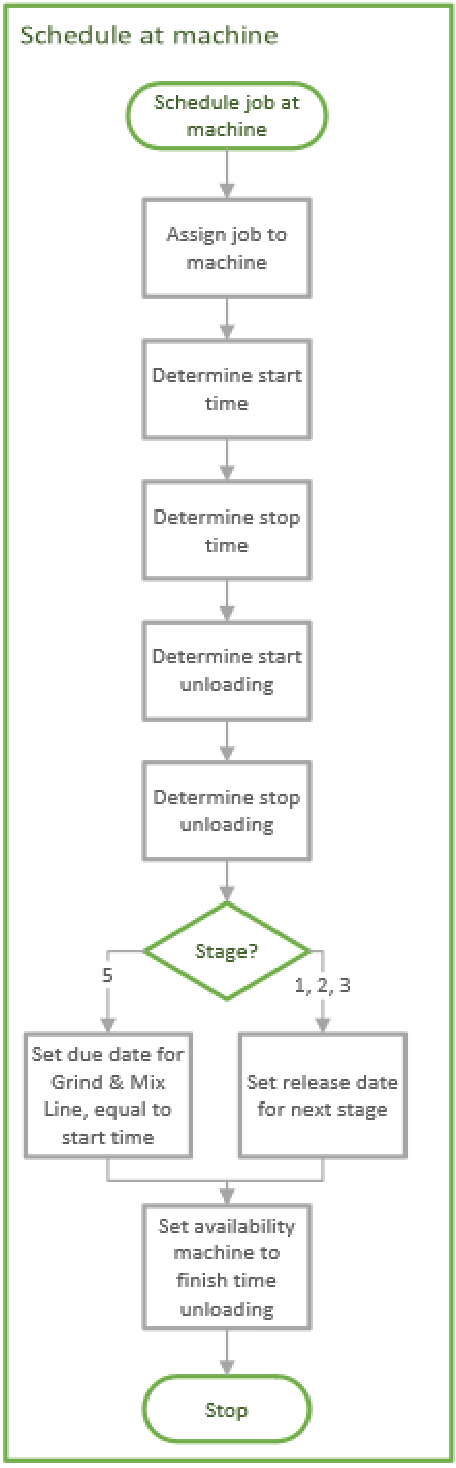

FIGURE 4.8: LOGIC FLOWCHART SCHEDULING AT MACHINE

The finish time of stage 3 and 5 is equal to the start time (S_{ijm}) plus the processing time (p_{ijm}) plus a setup time (sd_{ijkm}). At stage 3, we only have a setup time if we produce a job of a different product than the preceding job. This is because of the so-called 'nalooptijd'. Recall toy problem 1 and Section 2.4. At stage 5, the setup time represents the time it takes to replace the mould if the product requires a different diameter size. We included the start-up program (recall Section 2.4) into the processing time. Thereby, we use two different processing times for each product. One if the job is of the same product as the preceding job and one if the job is of a different product. In the latter case, the start-up program is included, and we have a somewhat larger processing time.

Start unloading

The time at which unloading starts at the stages of the grind and mix line is equal to the maximum between the time the next stage is available ($A_{i+1,m}$) and the time the job finishes minus a certain amount of time unloading can start before the job finishes ($F_{ijm} - rs_{im}$). Only for stage 1, we have a positive value for rs_{im}. Recall from toy problem 1 that unloading of stage 1 can start before processing finishes. For the other stages, the value for us_{im} is zero.

For stage 5, unloading starts when processing finishes. Recall that unloading is here defined as the time it takes to empty the cooling machine. In reality, the cooling machine releases product when its fill level is reached (Section 2.1). However, because we do not model the succeeding processes, we do not have to track this. Only the unloading of the cooling machine at the end is necessary to determine when the next job of a different product can start.

Stop unloading

Unloading finishes at the stages of the grind and mix line after the removal time (u_{ijm}) or a certain amount of time after processing of the job finishes ($F_{ijm} + rf_{im}$). Again, we take the maximum of the two. The second option only holds if unloading started before processing finishes. Recall from the start time of unloading that this is only possible for stage 1. In all other cases, we have a removal time that needs to be added to the start time of unloading. We need the second option to prevent unloading from finishing before processing finishes, which is impossible.

For stage 5, unloading finishes after the removal time. However, only if we need to unload, as discussed before. If we do not need to unload the cooling machine, the finish unload time is equal to the finish time of processing the job.

Release date next stage

At the grind and mix line, the release date for the next stage needs to be determined after scheduling the job at a machine. We use the release date to determine the start date in the next stage, as we have seen earlier this section. The release date (R_{ij}) of stage 2 and 3 is equal to the start date of unloading of the previous stage ($US_{i-1,jm}$). The release date of stage 4 and 5 is equal to the start date of stage 3 plus a certain amount of time ($S_{3jm} + ts_i$), recall toy problem 1.

As we discussed earlier this section, the first stage also gets a release date after scheduling the press lines. This release date is determined after planning the job at the silo. We elaborate on this later this section.

Due date grind and mix line

At stage 5, we set a due date for the grind and mix line. To be able to start the press in time, we need to finish the job at the grind and mix line in time.

Availability machine

Finally, we adjust the availability of the machine, such that we know for the next job when it can start. The machine is available after the job we planned finishes unloading.

Schedule job at silo

So far, we discussed how to schedule the jobs at the press and at the stages of the grind and mix line. In between, also the press silo is scheduled; recall Figure 4.7. Figure 4.9 shows a logic flowchart that illustrates how we schedule the press silo. We give each press silo a counter to store the amount of product and number of jobs of the same product that is scheduled so far at the silo. Using these counters, we know whether a job fits in the silo or we can determine its release date for the grind and mix line. Recall toy problem 2 for these relations.

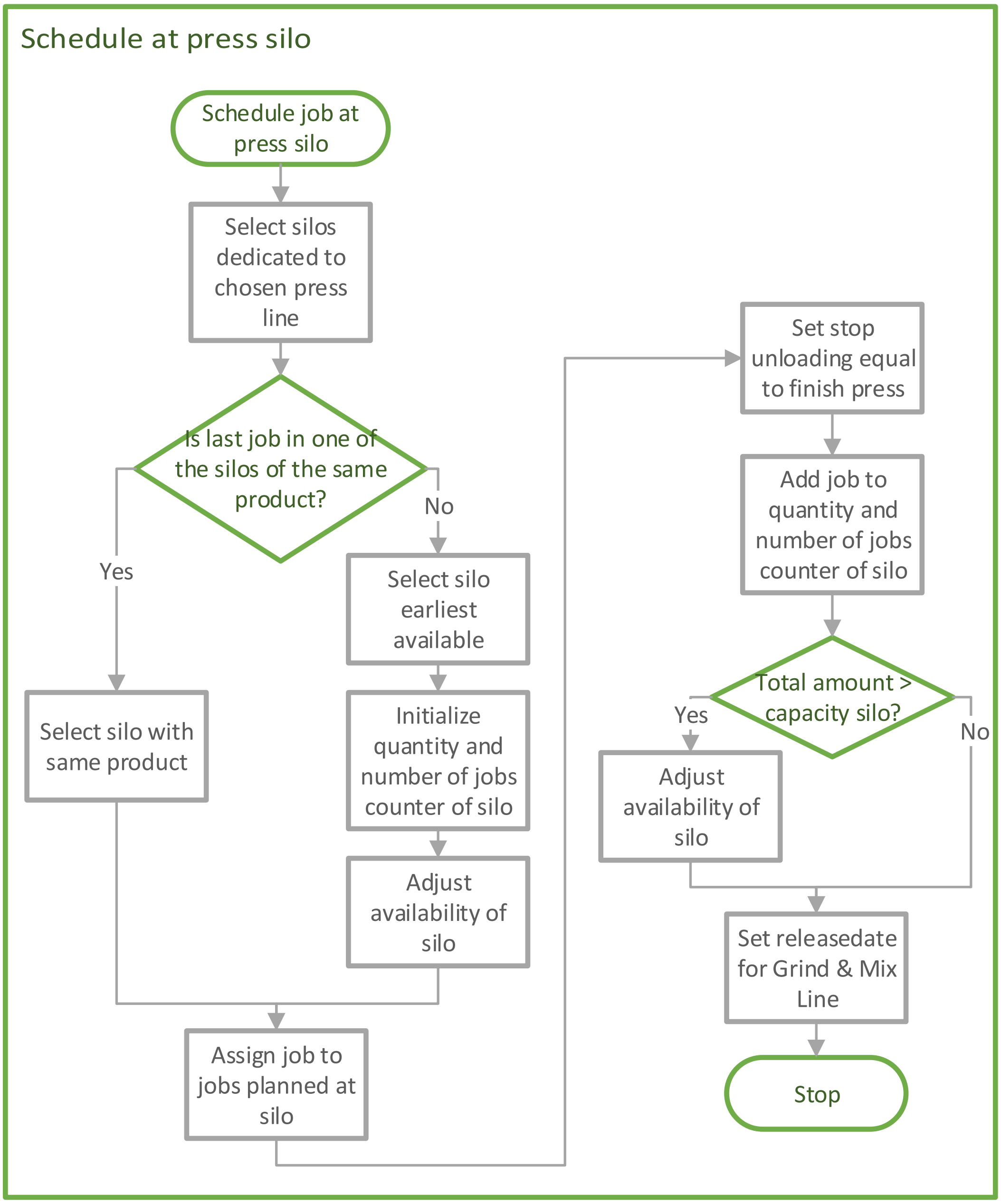

FIGURE 4.9: LOGIC FLOWCHART SCHEDULING AT PRESS SILO

We start to select one of the two press silos, dedicated to the press line. First, we check whether one silo contains the same product as the product of the current job. If so, we select that silo. If not, we select the silo that is earliest available. Recall from earlier this section that we only use one silo for a product. If we select the silo that is earliest available, we first reset the counters that store its content.

We know we have the first job of a new product, so the amount inside should be zero. We also adjust the time at which the silo is available (A_{im}) to the finish unloading time (UF_{ijm}) of the last job scheduled at the press silo.

Next, we assign the job to its silo and set the finish time of unloading equal to the finish processing time of the press. There are no accurate times for the press silos available in the MES. Therefore, we have to estimate from when to when we need the press silo. When the press finishes, we are sure that the job is also out of the silo.

The job and its amount are now added to the silo counters, where after we check whether it still fits in the silo when all jobs of the product would come at once. If it fits, we can determine its release date for the grind and mix line and continue with scheduling the next job at the press line. If not, there is only space for the job after one or more jobs finished pelletizing (recall toy problem 2). Therefore, we adjust the availability of the press silo. For the first job that does not fit, we know it fits after the first job of the product finishes pelletizing. So, we adjust the time at which the press silo is available to the finish unloading time of the first job. For the second job that does not fit in, we use the finish unloading time of the second job, and so on.

In both cases, we determine the release date for the grind and mix line. Therefore, we take the availability of the press silo (A_{im}) and subtract the processing time at the first two stages and the time difference (ts_i) between the start of stage 3 and the moment it enters the press silo from it. Thereby, the earliest moment the job can finish at the grind and mix line is when the silo is available.

In Figure 4.9, we do not set a start date for the job at the press silo. We first schedule the press lines. Thereby, we do not know when the job finishes at the grind and mix line. Therefore, we add the start dates after we scheduled the grind and mix line (see Figure 4.7). The start date at the press silo needs to be equal to the release date at the press silo, because the job cannot be stored elsewhere.

4.3.3 MAKE SCHEDULE FEASIBLE

With the algorithm explained so far, we have scheduled all jobs. However, the schedule may not be feasible, as we indicate at the start of Section 4.3. There are two initial states of the system possible. (i) The entire system is empty or (ii) one or more machines are still working on earlier scheduled jobs. The latter case is most likely because the plant runs for 136 hours per week. In that case, the probability of having a feasible schedule at once is larger because not all press lines can start at the same time; so, the grind and mix line has a chance to supply each press line of a new job in time. The differences between the time that each line becomes available for the new schedule are often large enough, such that the schedule is feasible at once.

When multiple press lines can start at the same time, we do not want to decide beforehand which line should start first and which line should wait. Therefore, we start to schedule at each line at the same time. In case we start with an empty system, we know the grind and mix line is not able to supply all lines in time. The schedule is always infeasible, and we have to make changes at the press schedule after we scheduled the grind and mix line. Because we are sure that we need to adjust our schedule, there is no need to correct the start date of the first job per line by its processing time for the grind and mix line. In case of an empty system, we can start each press line at time zero. When scheduling the grind and mix line, we get release dates for the press and press silo. We only have to change the start time of the assigned jobs such that it is equal to or larger than the release date. Of course, because we change the start time, we also have to change the finish date and the start and finish of unloading. In fact, we move the jobs in time in the press and press silo schedule.

We use this approach for both initial states of the system. For the second one, also infeasibilities can occur. For example, when still multiple press lines start at time zero. Another approach that we considered is to determine beforehand when each line should start by giving each line a different initial time from which it is available for the new schedule. An advantage of this approach is that the schedule is feasible in one iteration, because the grind and mix line can now supply each line with a job in time. A disadvantage, because of which we favour our approach, is that we do not know of which line we need to delay the start the most. Depending on the assignment to press lines and the objective of optimization, the line that should start first can be different. We only know this after scheduling the jobs at the press lines and still need to make some changes. Or we accept that the decisions about which line to delay are not optimal.

By our approach, the choice of which press line's start to delay most is based on the sequence of the first jobs at the grind and mix line. When creating a schedule, this sequence is based on the intermediate grind and mix line due date. The grind and mix line due dates are based on the start times of the jobs at the press. Thereby, the sequence of pellet jobs at the grind and mix line is equal to the sequence at the press lines and the line that starts with the job with the earliest due date starts first. When optimizing, by changing the sequence of the jobs at the grind and mix line, we see the effect on the objective value and the optimal decision will follow. Section 4.4 explains more about the optimization approach we use.

By making changes to the press schedule, the time at which the press silo is available changes for some jobs and so the release date for the grind and mix line. Thereby, we also need to change the start and finish dates at the grind and mix line to be sure the schedule is feasible. In some cases, this still results in an infeasible schedule. Therefore, we check whether the schedule is feasible after changing the grind and mix line schedule and if not, we change the start and stop times at all lines. This continues until the schedule is feasible or until a pre-determined amount of iterations. In this section, we order the jobs for the grind and mix line on their due date for stage 3; thereby, we always get a feasible schedule in one or a few iterations. Our optimization strategy (see Section 4.4) can make changes to the grind and mix line schedule that makes it impossible to find a feasible schedule. For example, when more than two press silos would be needed for one press line. Therefore, a limit to the number of iterations to find a feasible schedule saves us of ending up in an infinite loop of changing start and stop dates.

4.4 Optimization

Now we have a feasible schedule, based on an EDD dispatch rule (Section 3.2.2). However, we do not know whether it is a 'good' schedule. It may be worse than a schedule built on intuition of the operator. In Section 5.3 we determine the performance of our algorithms. We think that we can find a better schedule with more advanced heuristics. Recall Section 3.4 where we conclude that there are multiple divide-and-conquer heuristics or meta heuristics possible. We first explain the objective functions we use in our analysis. Next, we explain the optimization strategies we use to find a near-to-optimal schedule.

Objective function

To determine the performance of the optimization in Section 5.3 and to evaluate the performance of one schedule during optimization we need an objective function. For ForFarmers, it is important that jobs finish in time such that the farmer is delivered in time. Moreover, ForFarmers wants to use its production capacity as efficiently as possible. To fulfil both goals, we propose a multi-objective optimization. Finish of production in time is the main objective of our optimization. Therefore, we define three possible objectives:

- Total tardiness. Tardiness is the finish time of the job minus the due date of the job, insofar positive; otherwise it is zero. The total tardiness is the sum of the tardiness per job.
- Maximum tardiness. The maximum tardiness is the highest tardiness value of an individual job. The total tardiness can result in only one job that is very late. The maximum tardiness minimizes this maximum amount of time a job is late. As a result, more jobs can be a bit late.
- Number of jobs late. Another possibility is to minimize the number of jobs that finishes late. This objective minimizes the number of dissatisfied customers because of being delivered too late. However, the number of jobs that are late gives no information about the amount of time they are too late.

Each objective has its pros and cons. Using one objective, ForFarmers is interested in the value of the others. Making a trade-off on their own what is best for their current set of jobs. If there are multiple schedules with the same tardiness objective, we want to have the schedule that makes the most efficient use of the production lines. Therefore, we also define three make span based objectives:

- Make span (MS). The make span of a machine is the moment it finishes its last job. The make span is the make span of the machine that finishes last.
- Total make span (TMS). The total make span is the sum of the make spans per machine.
- Total make span press lines + total flow time grind and mix line (TMSPFT). For this objective, we take the sum of the make spans at the press and add the flow time of the jobs at the machines at the grind and mix line. We define the flow time as the amount of time a job spends at a machine. We want to minimize the flow time at the grind and mix line because it is cheaper for ForFarmers when the line is unused instead of used because of an inefficient schedule. By minimizing flow time, production takes a smaller amount of time and ForFarmers saves energy costs.

Optimization approach

We propose a combined approach that uses adaptive search to find a schedule for the press lines and simulated annealing to optimize the grind and mix line given the schedule for the press lines. There is a clear distinction between the grind and mix line and the press lines. Because the press lines are the bottleneck, we optimize them first. Given the schedule at the press line, we try to find the best grind and mix line schedule that optimizes the overall schedule. Figure 4.10 shows the same figure as Figure 4.6, with two added rectangles that indicate which steps are in the adaptive search or simulated annealing procedure. Note that we do not only optimize the grind and mix line with simulated annealing. We take the effect on the press schedule into account by making the overall schedule feasible, if possible (recall Section 4.3.3), and we search for the best overall feasible schedule. We explain the adaptive search and simulated annealing procedure later this section.

To illustrate our approach, we use an example. We have a set of jobs and we want to find the best schedule based on the make span objective. First, we use adaptive search to create a press line schedule in which the last job finishes as soon as possible (make span). We are not sure whether the grind and mix line is able to supply the jobs in time, just as in Section 4.3.3. Next, we use simulated annealing to find a schedule for the grind and mix line that does not change the make span that we found or that has the smallest increase possible. Therefore, each iteration of simulated annealing, a small change in the grind and mix line sequence is made, where after the schedule is made feasible if possible. Next, the objective value for the entire system is evaluated to determine whether the change results in a better schedule or not. Hence, if the change in grind and mix line schedule results in an increased finish date of the last job, the schedule is worse.

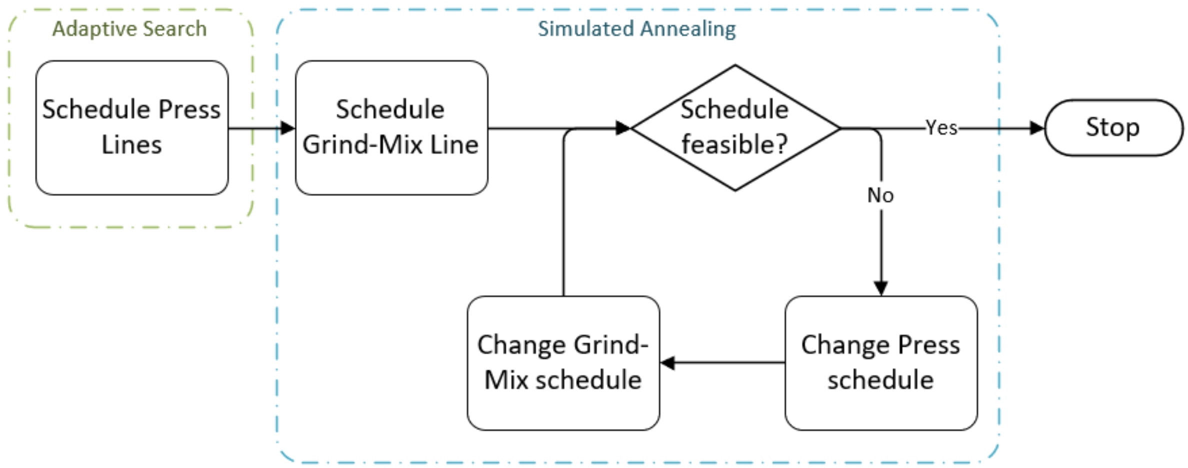

FIGURE 4.10: STEPS TO GET A FEASIBLE SCHEDULE WITH OPTIMIZATION

Before we explain the adaptive search and simulated annealing procedure, we first explain why we choose these optimization strategies. We concluded in Section 3.4 that divide-and-conquer heuristics or meta heuristics outperform simple dispatch rules. The production process at ForFarmers is highly specialized and it is especially challenging to model the connection between the grind and mix line and the press lines. Therefore, there is no specialized heuristic available and we designed our own dedicated algorithm.

To reduce the complexity of the problem, we divide the optimization in two subproblems. Thereby, there is a divide-and-conquer idea underlying our approach. A divide-and-conquer heuristic divides the problem in smaller subproblems and solves these subproblems one by one. The first subproblem contains the optimization of the press lines. The second subproblem optimizes the sequence for the grind and mix line, such that the overall schedule is optimized. Recall that we first schedule the press lines because the press silos have a limited capacity and because we want jobs of the same product and due date after each other at the press line (Section 4.3.1). An important reason for optimizing the press lines first is that the press lines are the bottleneck for ForFarmers in Deventer (Section 2.5). The grind and mix line has excess capacity and may be suboptimal, given an optimal press line schedule.

For the second subproblem, we choose to use the metaheuristic simulated annealing (Section 3.2.3). Metaheuristics are general procedures that can be used for many combinatorial optimization problems. We choose for a general procedure because of all the interrelations between stages that we need to consider. Furthermore, we choose metaheuristics because they are robust approaches that are able to find good quality solutions within reasonable time for realistic problem sizes (Aarts &

Lenstra, 1997). We use simulated annealing because it is a powerful tool that can escape bad, local optima. Each iteration, a small change to the current sequence is made and if the corresponding solution (neighbourhood solution) value is worse, it is still accepted according to a certain probability. Thereby, a much wider solution space is evaluated compared to methods which only accept changes that improve the objective value. In the end, still a local optimum is reached. However, because of all the worse solutions accepted and the wider solution space that is evaluated, the local optimum is likely to be close to the global optimum. Especially with the right parameter settings, which are explained later this section.

To construct a neighbourhood solution for the HFS several operations are possible. Most common operations are swaps and moves. With a swap, two operations in the schedule are interchanged. A move positions one operation somewhere else in the schedule. For the grind and mix line in our problem, we can move or swap the job in the sequence, because the sequence of jobs is the same in each stage. So, the operations for the grind and mix line are:

- Swap two jobs in the sequence of the grind and mix line.
- Move one job in the sequence of the grind and mix line.

If we would also use simulated annealing for the press lines, multiple operations are possible:

- Swap two operations in the sequence of one press line.
- Swap two operations between two parallel press lines.
- Move one operation in the sequence of one press line.
- Move one operation from one press line to another press line.

Each operation at the press line tears one batch of jobs of the same product apart, or merges it. Recall that we want to schedule the jobs of the same product as much as possible after each other to save time at the press lines. We could move or swap all jobs of the same product at once, but this is quite complex. We also must take care that we only swap between or move to lines that are allowed. Therefore, we use an alternative approach to keep jobs of the same product together.

To solve subproblem 1, we use adaptive search. Adaptive search chooses the next job to schedule with a certain probability instead of sorting the jobs and scheduling them one-by-one, as a simple dispatch rule does. Because the approach chooses the job with a certain probability, multiple iterations can be performed after which we keep the best schedule. We added a rule that schedules all jobs of the same product together. Advantage of adaptive search is that we still can spread the workload over the different press lines. To reach its full capacity, the plant in Deventer needs all press lines. If we use a total make span objective, it is reasonably optimal to schedule each job at the press line at which it is produced the fastest, in so far it finishes before its due date. If we use an approach like simulated annealing, this can result in a schedule in which a press line is not even used at all, after a lot of swaps and moves. Especially for sets of jobs with a lot of make-to-stock products with a due date multiple days in the future.

In the remaining part of this section, we explain the adaptive search and simulated annealing procedure.

Adaptive Search

To schedule the press lines, we use adaptive search (Kolisch & Drexl, 1996). Adaptive Search is a construction heuristic. It is a combination of a dispatching rule and random search techniques. Random search evaluates a certain amount of randomly generated schedules and advises the best one. Dispatch rules schedule the jobs in a certain order. So, the next job to choose when creating a schedule is fixed. Adaptive search constructs a schedule by scheduling the jobs one by one. Each job has a certain probability to be chosen next. Hence, the order is not fixed, nor totally random. To determine these probabilities, each job gets a priority. We prioritize the jobs based on their due date. So, their priority is equal to their due date. Next, the job gets a regret factor. The higher the regret factor, the more regret we have if the job is not chosen. The higher the regret factor of the job, the higher the probability of choosing the job to schedule next. The regret factor of a job is the difference between the maximum due date of the set of jobs and the due date of the job. The job that needs to be finished first gets the highest value.

$$Regret\ factor_j = \max_i d_i - d_j$$

The probability to be chosen next is based on the regret factor plus one to the power β, see formula below. β is the bias-factor (β ≥ 0). The higher the value of β, the higher the probability of choosing a job with a high regret factor.

$$Probability_j = \frac{(Regret\ factor_j + 1)^\beta}{\sum_i (Regret\ factor_i + 1)^\beta}$$

After scheduling the chosen job, we schedule all jobs of the same product consecutively at the same line. Next, we recalculate the priorities, regret factors and probabilities. We choose the next job based on the new probabilities. We repeat this procedure until we scheduled all jobs.

We construct n times a schedule for the press lines by this approach and keep the best schedule, given the used objective. In Section 5.3, we determine how many times we create a schedule and what values for beta we need. The creation of the press line with adaptive search is quite like the approach of Figure 4.7. The difference is that we take the next job with a certain probability, instead of taking the next job from a sorted list, and we construct multiple schedules to choose the best one.

Simulated annealing

Given the best schedule found for the press line, we continue with the grind and mix line. First, we create an initial schedule, following the same approach for creating a grind and mix line schedule as in Section 4.3. Next, we start simulated annealing. Figure 4.11 shows a flowchart for the simulated annealing procedure on a high level. When the neighbour solution value is equal to or better than the current solution value, the neighbour solution becomes the current solution. If the neighbour solution value is better than the best solution value, we remember it as the best solution found so far. When the neighbour solution value is worse than the current solution value, it becomes the current solution with a certain probability. This probability decreases over time. At the start, we accept almost every solution that gives a worse solution value. At the end, we almost only accept solutions with a better or equal objective value than the current solution.

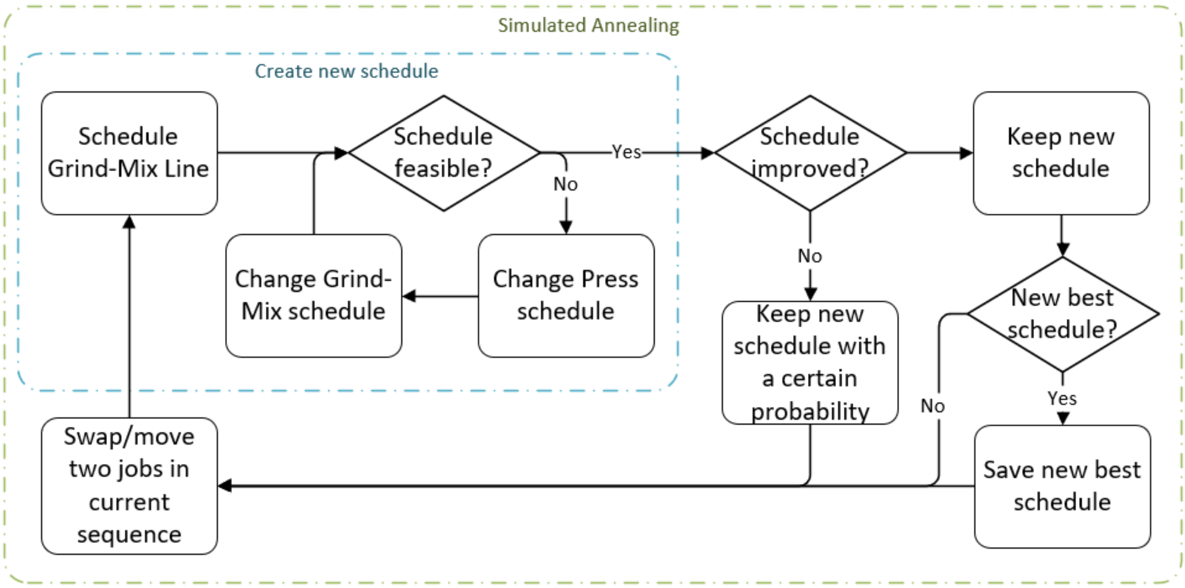

FIGURE 4.11: FLOWCHART SIMULATED ANNEALING

We start a new iteration by swapping two jobs in the sequence for the grind and mix line of the (new) current schedule, or by moving one job to another position in the sequence. Hence, we use a swap or move operation at the sequence of the grind and mix line. Move operators are important for lower temperatures, where a swap is more likely to result in a worse solution. The move operator is able to move a job to another position in the sequence. In Section 5.3.1, we see that for the flow time objective, it is optimal to cluster jobs of the same product. By a move operator we can move a job into a cluster, where a swap is more likely to also take a job out of the cluster or a preceding or succeeding cluster, which results in a worse solution. At high temperatures, this is no issue and the swap operator puts two jobs in another position. Thereby, we are able to evaluate whether it is useful to produce one job before the other.

Figure 4.11 shows that we again try to create a feasible schedule, as in Section 4.3.3, before we evaluate the solution value. Recall that we change start and finish dates such that no job starts at the press line before it finishes at the grind and mix line, to make the schedule feasible. Thereby, we solve a problem of our approach of starting all press lines together without knowing whether the grind and mix line is able to supply them. We want to take out this infeasibility to be able to compare the solution values when we run our algorithm. Otherwise, press lines start and finish too early and we get a better solution value than possible, because it is infeasible. Not in all cases it is possible to make the schedule feasible. For example, when the sequence at the grind and mix line is such that more than two press silos are needed, i.e. a job of the third product at a press line is produced before a job of the first product at the same press line. If after 50 iterations the schedule is still not feasible, we know it will not become feasible anymore. We do not accept it as the current or best solution and restore the current sequence to the previous current sequence. Running 50 iterations to try to find a feasible schedule takes only 15.8 milliseconds. If a schedule can be made feasible, we only need one or a few iterations. Hence, this takes less than one millisecond.

To reduce the number of times that we find an infeasible schedule after an operation, we limit the range in which a job can be moved or swapped. We call this additional parameter the 'neighbourhood width'. Because the assignment to and sequence at the press lines is fixed, we do not want to search in infeasible solution space. Jobs of the first product at a press line needs to be produced at the grind and mix line before jobs of the third product, otherwise they cannot be stored. By limiting the neighbourhood width, we decrease the probability of obtaining an infeasible solution by a swap or move operation. The neighbourhood width is set to 15, which means that when we randomly select a job, the second job must be within 8 jobs before or after this job in the sequence. For move operations, we also can only move the job 7 positions forward or backward in the sequence. Some trial-and-error experiments showed that 15 is a proper value. However, a somewhat smaller or larger range makes no big difference. We did not include the maximum number of iterations to get a feasible schedule, or the neighbourhood range in Figure 4.11 because we want to keep the figure simple, illustrating the basic idea behind our simulated annealing algorithm.

The probability that we accept a worse solution than the current one decreases over time. Simulated annealing uses some parameters to decrease this probability. These parameters also take care that the algorithm stops after a certain amount of iterations. Therefore, it uses a so-called cooling scheme, consisting of a cooling parameter/temperature (c), a start temperature (c_0), a stop temperature (c_{stop}), a decrease factor (α), and a markov chain length (k). The temperature starts at the start temperature and decreases until it reaches the stop temperature. When it reaches the stop temperature the algorithm stops. The temperature decreases by multiplying with the decrease value, which is a value between 0 and 1. The higher the decrease factor, the slower the cooling parameter decreases, and the more schedules are evaluated. The markov chain length represents the number of iterations that are evaluated at a certain temperature. Using the temperature (c), we calculate the probability of accepting a schedule with solution value B, and a current schedule with solution value A, by the following formula:

$$P_{AB}(c) = \begin{cases} 1 & B \leq A \\ e^{\frac{A-B}{c}} & B > A \end{cases}$$

Figure 4.12 shows some pseudo code that illustrates the cooling scheme. We will determine the value for the parameters in Section 5.3. The right values are important find a local optimum that is close to the global optimum.

```
c = c_0
WHILE c > c_stop LOOP {
        FOR i = 1 TO k LOOP {
                Swap two jobs
                Create feasible schedule
                Calculate acceptance probability
                Keep new schedule as current schedule with acceptance probability
                IF Solution value < Best solution value THEN
                        Best solution value = solution value
        }
        c = α * c
}
```

FIGURE 4.12: PSEUDO CODE COOLING SCHEME

4.5 Conclusion

This chapter explains how we model the process and what strategies we use to find a schedule. We model the process in five stages. The first stage consists of the sieve, hammer mill, and succeeding buffer 6. The second stage consist of the first mixer and the third stage of the buffer 7 combined with the second mixer. Stage four represents the press silos and connects the grind and mix line to the press lines. Stage five consist of the press lines.

The assumptions of the general hybrid flow shop problem (HFS) do not hold. There are all kind of different relations between stages. We illustrate these relations by two toy problems. Toy problem 1 goes into the relations that determine the start and finish time at a certain stage. Besides a processing time, stages also can have a removal time for unloading of the job. Unloading can start before the job finishes processing, when it finishes, or at a later moment in time. Important relations are:

- Stage 1, 2, and 3 can only start when unloading of the previous product finishes.
- Unloading of the hammer mill starts a certain time before the hammer mill finishes processing, but only if the first mixer is available. If not, it starts unloading when the first mixer becomes available.
- Unloading of the first mixer can only start when the mixer finishes processing.
- The second mixer can start immediately when unloading of the first mixer starts. Although, the succeeding elevator needs to be available.
- A certain amount of time after the first kilograms of the jobs went through the second mixer, they reach the press silo via the elevator transporting system. The press starts at this moment, if it is available. Otherwise, it is pelletizing a job from the second press silo. The job then starts when the press finishes and after the cooling machine is empty.
- When the press finishes, the cooling machine needs to unload. But only if the next job is of a different product.

Toy problem 2 goes into the connection between the grind and mix line and the press lines and the limited capacity of the press silos. It explains that the grind and mix line gets blocked when the job cannot be stored in the destination press silo. Therefore, when the press silo is full, the job needs to be planned to start at a later moment in time such that it finishes when there is space available in the press silo. Important assumption is that we assume that only one press silo per press line can be used for one product.

We designed our own algorithm to solve the problem, based on a divide-and-conquer strategy dividing the problem in two subproblems which are solved separately. In the first subproblem, we schedule the press lines with adaptive search. Adaptive search is a construction heuristic which constructs n times a schedule and keeps the best one given an objective function. Next for the second subproblem, we use simulated annealing to find an optimal sequence for the grind and mix line that optimizes the overall schedule; given the assignment to and order at the press lines by the adaptive search procedure.

Especially when the press schedule starts each press line at the same time, the grind and mix line may not able to supply each press line in time. Therefore, the schedule is not feasible after scheduling all lines. To make the schedule feasible, we change the schedule by changing the start and stop times per job at the press lines and grind and mix line, until the schedule is feasible.

Neighbourhood operations of simulated annealing at the sequence of the grind and mix line can create a sequence which is infeasible because it uses more than two press silos for one press line. Therefore, we stop trying to find a feasible schedule after 50 iterations. We also implement a neighbourhood width that limits the range in which two jobs can be chosen to be swapped, or one to be moved. If we select one job randomly, the other job should be within 8 jobs before or after the job in the sequence for a swap. For a move, the position to move to should be at most 7 places before or after the selected job.

We made one important assumption for the press silos, which connect the grind and mix line to the press lines. We assume that only one silo can be used for jobs on one product. In practice, both silos can be used. However, it is quite challenging to implement a rule that uses both silos. This decision only has small influence at the grind and mix line schedule.

5 PERFORMANCE OF SCHEDULING ALGORITHM

This chapter focuses on the research question: *What is the effect of the planning algorithm on the performance?* To test the performance of our algorithm and to provide insights for ForFarmers, we perform experiments in Section 5.3. To perform experiments, we define six test sets based on historical data in Section 5.1. Section 5.2 discusses validation and accuracy of our model. Finally, Section 5.4 provides the most important conclusions from this chapter.

5.1 TEST INSTANCES

To measure the performance of our algorithm and to perform experiments, we define multiple test sets. These test sets consist of a number of jobs in a certain time interval. We use test sets based on historical data. If we would experiment with current day production, by running our algorithm next to the decisions of the operator on intuition, the operator is more aware of the importance of his decisions. Thereby, he can think more carefully and make better decisions. The decisions made in the past, based on intuition, are not influenced by special attention for optimization. Therefore, historical sets give a good indication of daily performance. A disadvantage of historical test sets in our case is that we cannot trace back the original due dates. We use the time the job started to be loaded into the truck. However, we are not sure whether this is according to the original plan. The jobs finish processing before they go into the truck. Therefore, we are pretty sure that a zero-tardiness schedule is possible, unless our process time estimations are much larger than the original process times in the test set in particular cases.

To cover most characteristics of the production process, we define test sets covering different parts of the day and week. For example, make-to-order (MTO) products are mostly produced during the night and morning (recall Section 2.2). Whether the schedule starts with an empty system at the start of the week or connects to an existing schedule during the week is also an important factor to cover in different test sets.

Table 5.1 shows the test sets we use. The start and finish time indicate the interval at the grind and mix line that is included. Some jobs are pelletized at a later moment in time. Furthermore, we indicate the number of jobs to schedule and the number of batches of the same product for the press lines. Appendix C provides an overview of the times at which the machines are available for their first job.

TABLE 5.1: TEST SETS

Set	Start Time (Grind & Mix Line)	Finish Time (Grind & Mix Line)	# Hours	# Jobs	# Batches Press lines
1	18-02-2018 21:51:45	19-02-2018 06:03:02	≈ 8	77	17
2	25-01-2018 02:04:21	25-01-2018 13:41:31	≈ 11.5	86	13
3	26-01-2018 02:26:34	26-01-2018 15:25:28	≈ 13	85	15
4	03-02-2018 06:36:28	03-02-2018 15:28:49	≈ 9	60	13
5	09-01-2018 16:14:00	10-01-2018 07:43:00	≈ 15.5	128	17
6	25-01-2018 02:04:21	26-01-2018 01:08:08	≈ 23	171	24

Set 1 is a test set that involves the start of the plant for a new week (Sunday evening). Recall that the plant runs from Sunday evening until Saturday morning. Set 2 consist of an arbitrary Thursday morning and Set 3 of a Friday morning. Set 4 represents the last shift of the week at Saturday. Set 5 consist of an evening and night. Set 6 is a large set of almost one day, to test the performance over a larger period of time.

5.2 Model validation

Before we start experimenting, we discuss validation of our model. We want to know whether our model represents reality well. Therefore, we compare model output with real production data. We want to know whether the relations between stages are correct. To test the relations, we compare the original schedule with model output, using the same sequence and process times as in the original schedule. We expect our schedule to start jobs before or when unloading of the preceding stage starts. Therefore, jobs should start at a stage before they finish at the preceding stage, recall Section 4.2. Because we use the same processing times, we expect jobs to be processed at more or less the same time as in the original schedule. Unfortunately, we can only perform this validation for the grind and mix line stages. For the press lines, our model schedules per job. The original schedule consists of larger batches that combine a number of jobs. We do not have process times per job at the press lines. Hence, we cannot compare with the model output.

To validate whether the press lines are scheduled according to reality, we compare the original schedule to the model output, using the same sequence but the estimated process time. We expect that short before stage 3 finishes, the job enters the press silo. The last job of a product should leave the press silo a certain amount of time before the press finishes, because when the cooling machine unloads, the press silo does not contain the same job anymore. Furthermore, we expect larger differences at the grind and mix line because we now use estimated average process times compared to the original process times.

For validation, we use Set 1. First, we validate the grind and mix line. Figure 5.1 shows the original and model schedule for Set 1. Three machines have a '(o)' behind their name. These rows show the schedule according to the exact process data, the original schedule. The rows below them show the schedule according to our model, with the same order in which the products are produced and the same processing times. Figure 5.1 shows that for the first two hours, the model output is equal to the original schedule. Over a period of 8 hours, we gain 12 minutes on the original schedule. A small discrepancy, we can explain this by two main reasons. For the grey jobs half way the schedule, there is a gap at buffer 7. We do not know why this gap occurs in the original schedule. Our model output does not have this gap and gains on the original schedule. The same holds for the last, orange, jobs. In the original schedule, these jobs start later. In the model output, it starts directly after the red jobs.

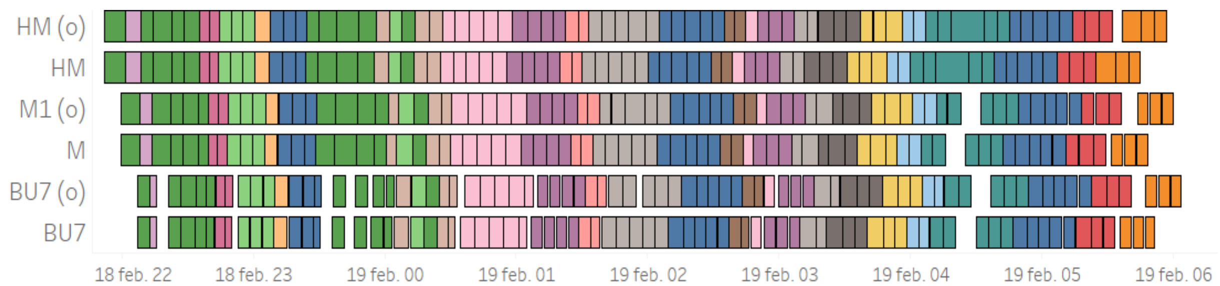

FIGURE 5.1: GANTT CHART VALIDATION GRIND AND MIX LINE (HM = HAMMER MILL, M1 = MIXER 1, BU7 = BUFFER 7/ MIXER2)

Figure 5.2 shows a close-up of the first jobs at the machines of the grind and mix line, to increase the readability of the times that a job starts or finishes. The start/stop relations of our model are also according to reality. The bars of our schedule are in almost the same position as the bars representing the true production data.

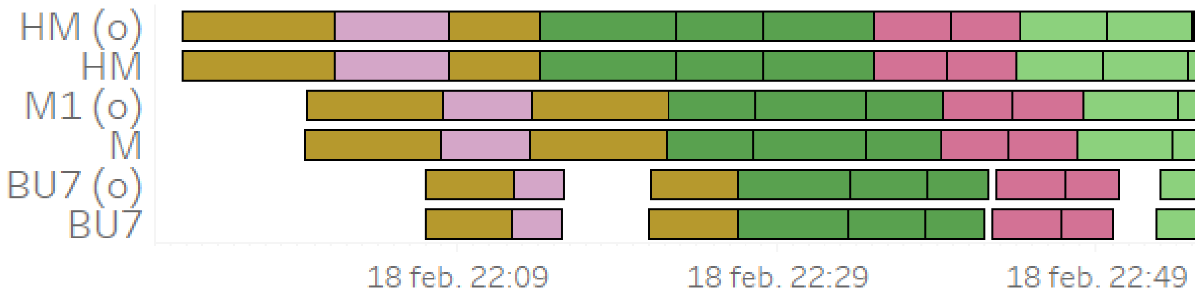

FIGURE 5.2: GANTT CHART VALIDATION GRIND AND MIX LINE FIRST JOBS

Figure 5.3 shows the original '(o)' schedule based on true production data compared to the schedule constructed with our model. We choose to only include press line four and its press silos to illustrate the most important relations and to keep a clear picture. Each colour represents a product, each rectangle a job. We highlighted the jobs that are produced at press line four with a green or blue colour. Each colour represents jobs of the same product. We have no data for the press silos of the original schedule; so, for the original schedule no press silo is shown. For the press lines, the original schedule only has start and finish dates of the entire batch. Therefore, the press lines of the original schedule have one large bar per product, versus shorter bars per job of our schedule.

During the first hours, our model gains on the historical schedule at the grind and mix line. The first cluster of blue jobs is produced earlier in our schedule than in the original schedule. Some jobs in the original schedule had a failure and took some more time. Halfway our grind and mix schedule, there is a gap. The next job cannot be produced earlier because there is no press silo available (not shown in the figure). Thereby, our schedule recurs to the original schedule.

Figure 5.3 shows that the press line starts shortly after stage 3 started. The original schedule shows that the press line starts before the buffer 7/ second mixer stage finishes. Hence, we implemented this relation successfully. The press silo also shows that the first job of a product enters it before or when the press starts. As we expected, the press silo is empty a certain amount of time before the press finishes, because the cooling machine needs to unload.

The impact of our assumption to only use one press silo for jobs of the same product (Section 4.3) becomes clear by Figure 5.3. In the first cluster of five blue jobs at the grind and mix line, 30,000 kg is produced. Thereby, press silo 308 is full. Later in the schedule, we see that the second cluster of five jobs in the original schedule is split up in our schedule. At the moment that the first two jobs of the second cluster are produced, they fit into the press silo. The press finished at that moment already two jobs. Thereafter, we produce each job when it fits in the press silo, just when a job finishes at the press. In the original schedule, silo 307 is used to store the second cluster of five blue jobs. Our control rule that only uses 1 press silo for jobs of one product works correctly. However, it causes large deviations at the grind and mix line when we schedule jobs in the same sequence as in the original schedule. The make span per machine of the grind and mix line is much larger than for the original schedule, as Figure 5.3 shows.

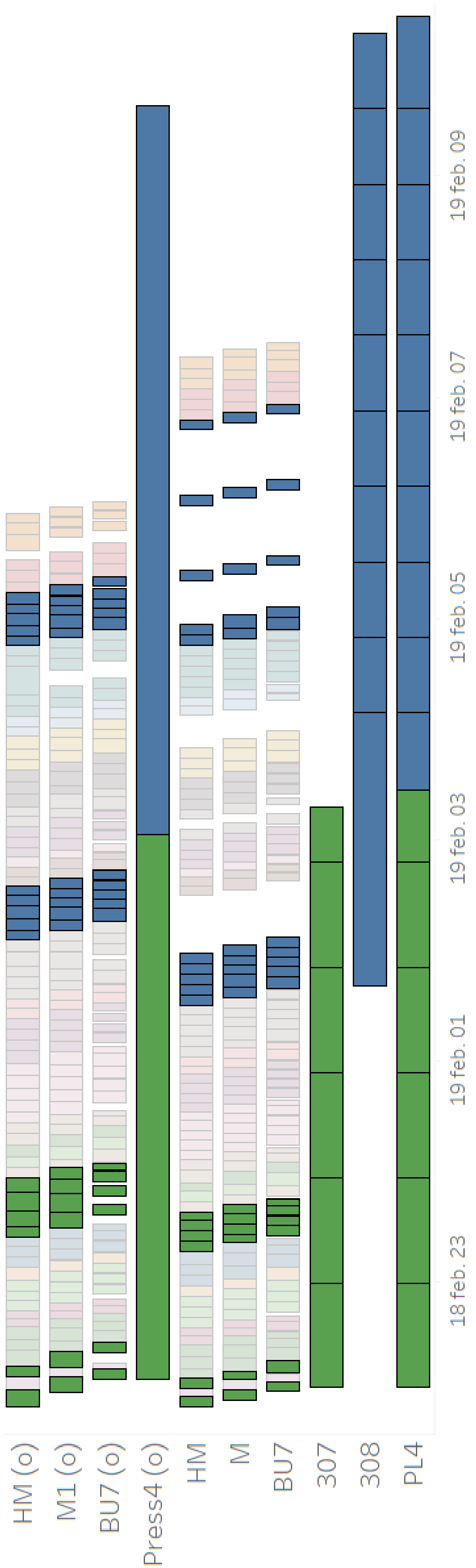

FIGURE 5.3: GANTT CHART VALIDATION PRESS LINES (HM = HAMMER MILL, M1 = MIXER 1, BU7 = BUFFER 7/ MIXER2, 307/308 = PRESS SILO, PRESS4/PL4 = PRESS)

5.3 EXPERIMENTS

To test the performance of our algorithm and to provide insights for ForFarmers we perform multiple experiments. First, we want to know the performance of our algorithm. How much time do we save by scheduling using our algorithm? Section 5.3.1 explains the improvement. We discuss the different objectives and their performance improvement. Next, we investigate the improvement potential at only the grind and mix line, in Section 5.3.2. Section 5.3.3 provides a sensitivity analysis to show the sensitivity of the parameters of the algorithm. Section 5.3.4 tests the optimal solutions found on feasibility if we include the (transport to) finished product silos.

Our algorithm consists of multiple parameters, recall Section 4.4 for their purpose. For simulated annealing (SA) we use a start temperature (c_0), a stop temperature (c_{stop}), a decrease factor (α), and a markov chain length (k). For adaptive search (AS), we use the bias-factor beta (β) and the number of iterations. Table 5.2 shows the values we use for the parameters in our experiments. Appendix D explains how we determine these values.

TABLE 5.2: PARAMETERS OPTIMIZATION ALGORITHM

Parameter	Value
SA - Start temperature (c_0)	20,000
SA - Stop temperature (c_{stop})	1.5
SA - Decrease factor (α)	0.99
SA - Markov chain length (k)	600
AS - Bias-factor (β)	Max (3; #Jobs – 150)
AS - Number of iterations	10,000

We focus at the make span objectives instead of tardiness objective in this section, because the tardiness is most likely to be zero in the experiments (see Section 5.3.1). In Section 4.4, we defined three make span objectives: (i) the make span (MS), (ii) the total make span (TMS), and (iii) the total make span of the press lines plus the flow time of the jobs at the machines of the grind and mix line (TMSPFT). The make span objective minimizes the time that the last job finishes. Therefore, the time each press line finishes comes closer together. Recall that jobs are combined in (large) batches and that processing times differ per press line. Because jobs are produced in (large) batches of the same product, there can be large differences between the time that each press line finishes. In our test sets, this is also the case. Therefore, the make span is poor, and there is a lot of optimization potential for the make span objective. To reduce the make span, jobs are moved to other press lines, which likely takes more time to produce them. As a result, we need more time to produce the same amount of product, compared to the total make span objective. The TMS objective has a larger tendency to produce jobs at the press line at which they are produced the fastest. Therefore, there can be large differences between the make span per press line, but the sum is minimized. In Section 5.3.1, we explain more about the differences between the make span and total make span.

For the TMS and TMSPFT objective, the objective consists of the sum of two parts, the total make span at the press (stage 5) and the total make span at the stages of the grind and mix line, or the flow time of the jobs at the machines of the grind and mix line (stage 1, 2, and 3). The press silos of stage 4 are not included in the objective function. With the TMS objective, release dates caused by the press schedule (recall toy problem 2) can limit the optimization potential. The grind and mix line schedule with minimum total make span can still have gaps and there is less urgency to reduce waiting times between stages (recall toy problem 1). In most of the cases, reducing waiting times does not influence the make span of the machine, because the last job cannot be produced earlier in time because of its release date for the grind and mix line. Reducing waiting times always reduces the TMSPFT objective. Therefore, we expect larger improvements for the latter objective.

5.3.1 PERFORMANCE OF ALGORITHM

We compare the performance of our algorithm on the different objectives with the objective values of the original schedule. To determine the objective value of the original schedule, we use our model to schedule the jobs in the original sequence. We choose not to use the original data to find the original objective value because there is already a deviation between the original data and our model with process time estimations. Using an original objective value based on process time estimations makes the comparison with the optimization objective values more reliable, because the optimization algorithm makes use of the same process time estimations. The first part of this section shows how much we can improve the performance. The second part provides insights in how the best schedules looks like.

Table 5.3 shows that for each test set, except Set 4, a schedule with zero tardiness is found. For Set 4, one job finishes late in the optimal solution. The estimated average process time of this job and of the preceding jobs is larger than the original process time. The job also can be produced at two lines only. Thereby, the job cannot finish before its due date.

TABLE 5.3: TARDINESS OBJECTIVE VALUES PER TEST SET

Set	Total Tardiness (sec)	# Jobs late
1	0	0
2	0	0
3	0	0
4	985	1
5	0	0
6	0	0

Table 5.4 shows the objective value improvements for different make span objectives and test sets, when we compare our algorithm to the original schedule. For the make span objective, we save 5% to 11%. On average this results in a make span reduction of 8.2%. For our test sets, this results in an average time reduction of 94 minutes. The larger the period for which we create a schedule, the higher the possible time reduction. Especially for larger sets of jobs, with a lot of make-to-stock jobs with a due date later in time, there is more improvement potential because these jobs can be produced in any order. For the total make span only a reduction of 2.6% is possible, and for the TMSPFT objective a reduction of 4.8%.

TABLE 5.4: IMPROVEMENTS OF OPTIMIZATION FOR DIFFERENT OBJECTIVES

Set	Make Span (MS)		Total Make Span (TMS)		Total Make Span Press + Flow Time (TMSPFT)	
	Minutes	*%*	*Minutes*	*%*	*Minutes*	*%*
1	62	7.9%	68	1.3%	164	3.4%
2	44	5.1%	732	11.5%	308	5.7%
3	90	8.7%	237	3.5%	276	4.5%
4	55	7.1%	307	6.1%	106	2.4%
5	113	9.2%	-872	-10.6%	245	3.3%
6	202	11.1%	435	3.6%	1011	9.3%
Average	94	8.2%	151	2.6%	352	4.8%

TABLE 5.5: IMPROVEMENT TOTAL MAKE SPAN OF PRESS ONLY

Set	Total Make Span Press	
	Minutes	*%*
1	190	5.3%
2	226	5.6%
3	203	4.3%
4	46	1.4%
5	168	3.1%
6	889	11.0%
Average	287	5.1%

Table 5.5 shows the total make span reduction for the press only, the total make span of the press lines can be improved by 5.1%

At the end of the introduction of Section 5.3, we predict that the TMSPFT objective would give higher improvements as the TMS objective. Table 5.4 shows that this is true. The TMSPFT objective has an average reduction of 4.8%. The TMS objective has a reduction of only 2.6% on average. The reduction for the TMS objective differs a lot per test set. From Set 2, which has a total make span reduction of 11.5% to Set 5 with an increase of 10.65%. Table 5.5 shows a reduction for the total make span of the press lines of Set 5. Therefore, we conclude that the total make span at the stages of the grind and mix line is far worse than for the original schedule. This is possible because of the release dates for the grind and mix line caused by the press schedule (recall toy problem 2). For e.g. Set 1, we also see that the improvement in total make span is smaller than in the total make span of the press only. Hence, the total make span of the stages of the grind and mix line is again larger than in the original schedule.

In the remaining part of this section, we provide more insights in how the best schedules for different objectives look like. Figure 5.4 shows four schedules for Set 1. The first one is the original schedule, the second one the best schedule according to the make span objective, the third one the best schedule according to the total make span objective, and the fourth one the best schedule according to the TMSPFT objective.

Figure 5.4 shows large differences in make spans per press line for the TMS and TMSPFT objective. Recall from Section 2.1 that line 4 and 5 are lines with an expander, which can produce at a higher speed. It is better to produce more jobs at these lines to minimize the total make span. For a total make span objective at the press lines, it is optimal to produce each job at the fastest line where it is allowed to be produced, in so far each job finishes in time. Thereby, it might be theoretically optimal to not use a ‘slow’ press line. In practice, we also want to use each line to produce as much as possible. We solve this by our adaptive search scheduling approach that schedules job-by-job at the earliest available, allowed press line (Section 4.4). Each iteration we create a different schedule and we keep the schedule with the minimal total press make span. The make span objective minimizes the time that the last job finishes. The time when the last job at the other lines finishes is thereby not that important. Figure 5.4 shows that all press lines finish more or less at the same time. Where the total make span objective starts each line as soon as possible, the make span objective does not have this tendency. For example, line 2 in Figure 5.4 can start at any time, in so far its last job finishes before the last job of the entire set.

Figure 5.4 also shows the difference between the optimal grind and mix line schedules. Remember that the press lines are the bottleneck for ForFarmers in Deventer and the grind and mix line has excess capacity. Therefore, all schedules contain gaps at the grind and mix line. For all schedules, the grind and mix line is able to supply the press lines in time. In the schedule for the TMS objective, the last job cannot be produced at an earlier moment in time because the press silo is not available. Therefore, there is no urgency to optimally schedule the preceding jobs, if they are ready in time before they need to start at the press line. Any change in the grind and mix line schedule does not change its make span because the make span is fixed by the job that cannot be produced earlier. We cannot identify a pattern in the grind and mix line schedule for the MS and TMS objective, both look quite random.

The schedule for the TMSPFT objective shows that if we want to optimize the flow time of the jobs at the grind and mix line, we save time by sequencing jobs of the same product next to each other. Of course, in so far the succeeding press silo can store these jobs. Again, we see gaps in the schedule because of the excess capacity of the line. Scheduling of jobs of the same product next to each other saves time because we save the setup time at stage 3, the buffer 7/ mixer 2 stage (Section 2.4, 4.2). Interesting to see is that the first jobs are not clustered. Recall that test set 1 is the only set that starts with an empty system. Therefore, to minimize total make span of the press lines, the lines need to start as soon as possible and should not stop intermediately because they have no more jobs waiting in the press silo. After some time, enough jobs are waiting in silos and jobs of the same product are scheduled next to each other. When we start with an occupied system, the grind and mix line can first produce multiple jobs for the press line that is available first, then for the press line that becomes available second, and so on.

Based on improvement in minutes for the total make span (TMS) in Table 5.4 and the total make span of the press in Table 5.5, we see that for three test sets, Set 1, 5, and 6, the improvement in minutes of the total make span is smaller than the improvement in minutes of the total make span of the press. Figure 5.4 shows that for Set 1, the make span of the grind and mix line for the TMS objective is indeed worse than for the original schedule. If the total improvement is smaller than the improvement for the press only, then the grind and mix line schedule for the TMS objective performs worse than the original schedule. Hence, the improvement for the make span of the grind and mix line is negative and the improvement of the total make span smaller than the improvement of the total make span of the press only. For the TMSPFT objective, we see for one test set, Set 1, that the TMSPFT improvement in minutes is smaller than the improvement of the total make span of the press lines. Hence, for Set 1, the flow time at the grind and mix line for the TMSPFT objective is worse than in the original schedule. We can see this in Figure 5.4 where the jobs of the same product are more clustered for the original schedule than for the optimal schedule of the TMSPFT objective, especially at the start of the schedule.

ForFarmers wants the press lines to run as much as possible, because they are the bottleneck and determine how much the plant can produce. The TMS and TMSPFT objective both give schedules that start the press lines as soon as possible. For the MS objective this is not necessarily the case. TMS and TMSPFT have a larger tendency to produce jobs at a line where they are produced the fastest. Therefore, it is likely to give a schedule that produces the same amount in less time. However, for the next set of jobs, the faster lines with an expander (line 4 and 5) are available the latest (see Figure 5.4). Therefore, for the next set, more jobs need to be produced at the ‘slower’ press lines. We do not know whether this is better than a MS objective in the long run. With the MS objective, each line is available more or less at the same time for the next set of jobs. The MS objective is especially useful for the last shift of the week when operators have to stay until the last press line finishes.

At the grind and mix line, the TMSPFT objective is the only objective that really optimizes the sequence in our case. Jobs of the same product are clustered to reduce the flow time. The make span objective depend on the press lines. The grind and mix line schedule does not influence it, in so far it supplies the press lines in time. Therefore, the make span objective is not that useful for the grind and mix lines. The TMS objective should also minimize the make span at the grind and mix line. However, Figure 5.4

shows that because the last job cannot be produced earlier, it loses its power. The sequence before the last job can be any sequence, in so far it supplies the press lines in time. In the next section, we investigate the different objectives if we ignore the press lines to provide insights in their behaviour.

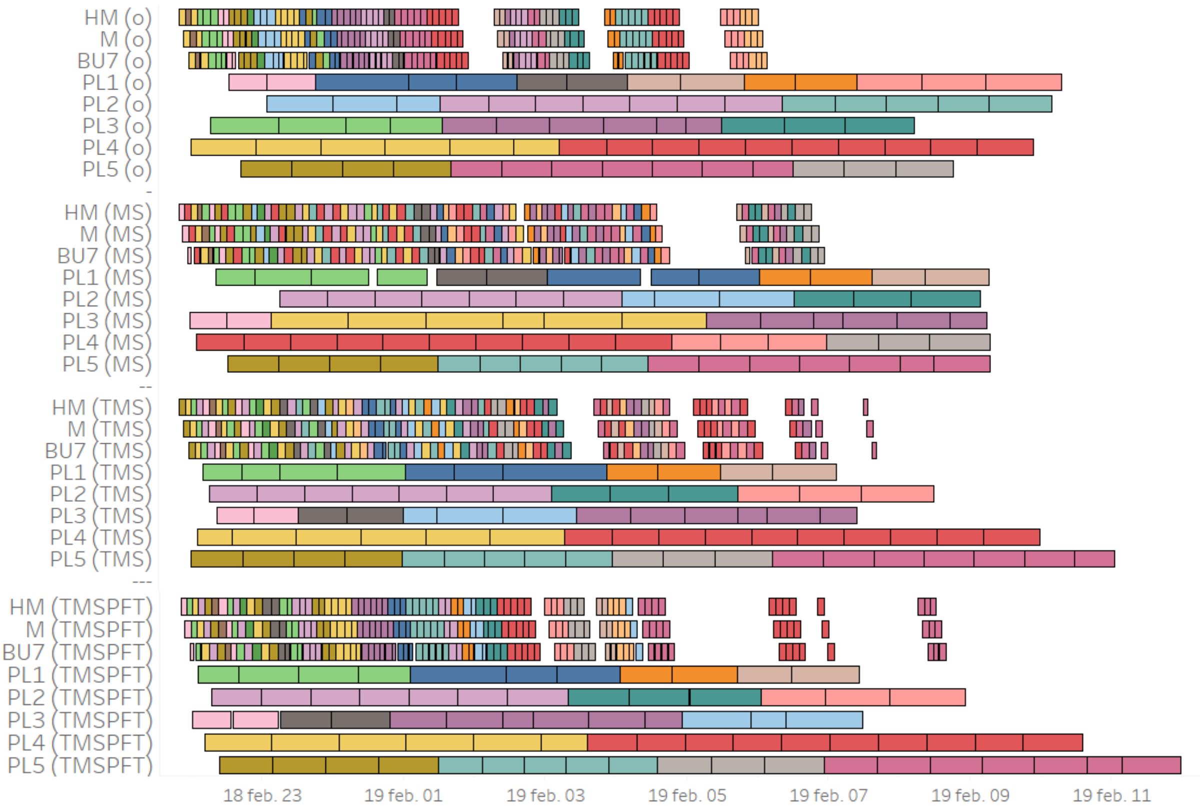

FIGURE 5.4: OPTIMAL GRIND AND MIX LINE SCHEDULE PER OBJECTIVE FOR SET 1 (HM = HAMMER MILL, M1 = MIXER 1, BU7 = BUFFER 7/ MIXER2)

5.3.2 Optimization Grind and Mix line

In Section 5.3.1, we discuss that because the grind and mix line is not the bottleneck, its schedule can be random for the MS and TMS objective, insofar due dates are met. We therefore choose to add the third objective, taking the flow time at the stages of the grind and mix line, to optimize at least something at the grind and mix line. In this section, we investigate the grind and mix line schedules for the different objectives when they are not influenced by succeeding stages. Therefore, we only schedule and optimize the first three stages and assume that succeeding press silos are always available. First, we compare the schedules for the different objectives. Second, we remove the setup time in stage 3 and try to identify a pattern. We expect that it is not necessary anymore to cluster jobs of the same product without the setup time.

Figure 5.5 shows four schedules for the grind and mix line of Set 1. The first schedule shows the schedule when jobs are scheduled in the original sequence. The second, third, and fourth schedule show the optimal schedule for the different objectives. Figure 5.5 shows that the optimal schedules are slightly better than the schedule representing the original sequence. Because operators of ForFarmers often already cluster jobs of the same product, the improvement is small. For the MS and TMS objective, there is now an incentive to sequence the jobs optimally. Clustering jobs of the same product reduces the (total) make span at the grind and mix line.

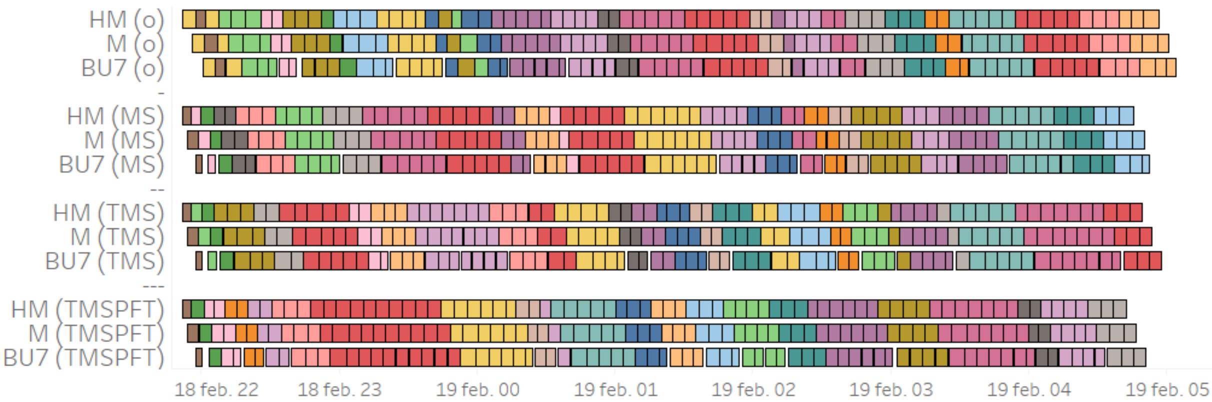

FIGURE 5.5: GRIND AND MIX LINE SCHEDULE, ONLY GRIND AND MIX LINE, FOR SET 1 (HM = HAMMER MILL, M1 = MIXER 1, BU7 = BUFFER 7/ MIXER2)

Figure 5.5 shows that it is optimal to cluster jobs of the same product at the grind and mix line. An important reason is the setup time at the third stage, the buffer 7/ mixer 2 stage. When the next, adjacent job is of another product, the second mixer must wait for the elevator transporting system to be available before it can start (Section 2.4 and 4.2). By clustering jobs of the same product, we save setup times. Figure 5.6 shows the optimal TMSPFT schedule of Figure 5.5 compared to the optimal flow time schedule if there are no setup times at stage 3, e.g. when there would be a second elevator transporting system. Without these setup times, there is no incentive to cluster jobs of the same product.

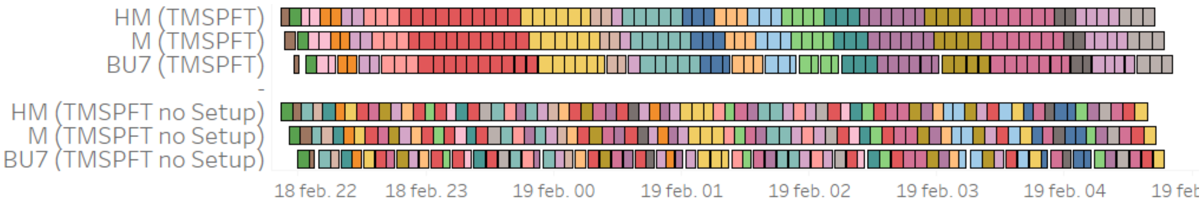

FIGURE 5.6: GRIND AND MIX LINE SCHEDULE SET 1 WITH AND WITHOUT SETUP TIMES IN STAGE 3 (HM = HAMMER MILL, M1 = MIXER 1, BU7 = BUFFER 7/ MIXER2)

The optimal schedule without setup times at stage 3, in the figure above, looks quite random. We did not find a pattern. For most jobs, the hammer mill stage is the bottleneck, i.e. takes the most time at the grind and mix line. Therefore, it does not have to wait at the succeeding stages. When a job finishes at the hammer mill, the preceding job at the succeeding stages already finished. So, the machine of the next stage is already available. Therefore, their sequence does not influence the flow time.

In the remaining part of this section, we provide some analysis on the waiting times at the stages of the grind and mix line, caused by jobs that has to wait before they can move to the next stage (recall toy problem 1, Section 4.2).

When optimizing the grind and mix line, we minimize the waiting time between stages. Recall that if a machine cannot unload it has to wait (Section 2.4 and 2.5). Table 5.6 shows the total waiting times per stage of the grind and mix line for Set 1 and 2. The waiting times are the highest for the original schedule. For the optimal TMSPFT schedule, the waiting times reduce by 44% and 28%. For the optimal schedule without setup times at stage 3, the waiting times even more reduce, with 72% and 86% compared to the original schedule. Thereby, we conclude that the setup times at stage 3 cause waiting time at the preceding stages. When the setup time can be reduced, it increases the utilization of all three stages.

TABLE 5.6: TOTAL WAITING TIMES AT GRIND AND MIX LINE (TIMES IN SECONDS)

	Set 1			Set 2		
	Original Schedule	Optimal Flow time	Optimal without setup	Original Schedule	Optimal Flow time	Optimal without setup
1. Hammer Mill	577	47	0	893	421	27
2. Mixer 1	1992	1392	724	2508	2024	442
3. Buffer 7	0	0	0	0	0	0
Total	2569	1439	724	3401	2445	469
% Reduction		44%	72%		28%	86%

Most waiting time is at the second stage, the first mixer. Hence, this mixer finishes multiple times before the second mixer finishes, even if there are no setup times for the second mixer. Figure 5.7 shows that the smaller jobs cause the waiting time at the first mixer. The smaller the quantity of the job, the more possible waiting time. Jobs of 6,000 kg never have to wait in the optimal schedule without setup times of Set 1, because the hammer mill has the largest processing time. Hence, if a job finishes at the hammer mill, the preceding job already finished at the succeeding stage. A job with a smaller quantity has to wait if it is produced after a large job. The large job takes more time at stage 1 and 3, of which the processing time depends on quantity. The process time at the second stage does not depend on quantity. Hence, the small job takes the same amount of time as the large job. The small job finishes earlier at the preceding hammer mill stage. So, mixing 1 can start earlier and finishes before the second mixer is available.

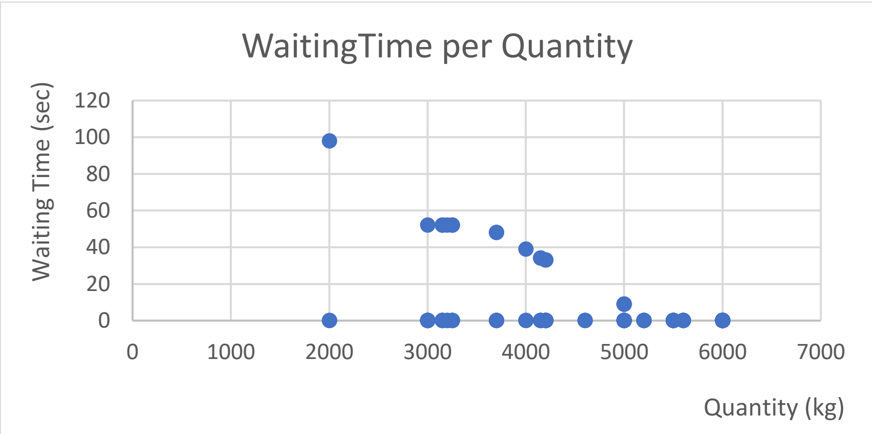

FIGURE 5.7: WAITING TIME PER QUANTITY, TEST SET 1 OPTIMAL SCHEDULE WITHOUT SETUP TIMES

5.3.3 Sensitivity Analysis Parameters

The quality of the solution depends on the parameter settings (Table 5.2). Appendix D explains how we determine the values for these parameters for our analysis. In our analysis, our focus is on finding the improvement potential of our algorithm compared to traditional scheduling. Therefore, running times are not that important. Although the settings used in our algorithm already give a short running time of 2 to 4 minutes, we want to show in this section the effect of shorter running times on the solution value. We first analyse whether the TMSPFT objective value becomes worse when we decrease one of the simulated annealing parameters, which saves running time. Next, we discuss a sensitivity analysis on the total tardiness objective value. Because our algorithm can easily find a zero-tardiness schedule for most of our test sets, we reduced the due dates by 50%.

We reduce the running time by reducing the parameter value of the decrease factor, markov chain length, and/or start temperature. We choose not to reduce the number of iterations to find a press schedule. The 10,000 iterations we use only take 13 to 22 seconds. Reducing this number reduces the probability of finding a near-to-optimal solution against only small time savings. The simulated annealing part of our algorithm takes more or less 120 seconds, depending on the problem size. We chose these parameters relatively high to be on the safe side. First, we analyse the solution value and running time for smaller start temperatures. Next, we look at the solution value and running time for smaller decrease factors, or markov chain lengths. We only reduce one parameter at once to test its behaviour. We use Set 1 to 4 for our sensitivity analysis. We use the TMSPFT objective for our analysis, because we are certain that simulated annealing optimizes the sequence at the grind and mix line for this objective (recall Section 5.3.1).

Figure 5.8 shows the solution value and running time for different values of the start temperature for the TMSPFT objective. In Appendix D, we choose the start temperature quite high, to be on the safe side. Figure 5.8 shows that lower start temperatures still results in a good solution. According to Appendix D, 5000 is the lowest start temperature in the range we are looking at (see Appendix D for explanation). Start temperatures lower than 5,000 weaken the power of simulated annealing, although they result in the best solution for our test sets. We save more or less 20 seconds of running time by using 5,000 as start temperature. Hence, lowering the start temperature has a limited effect.

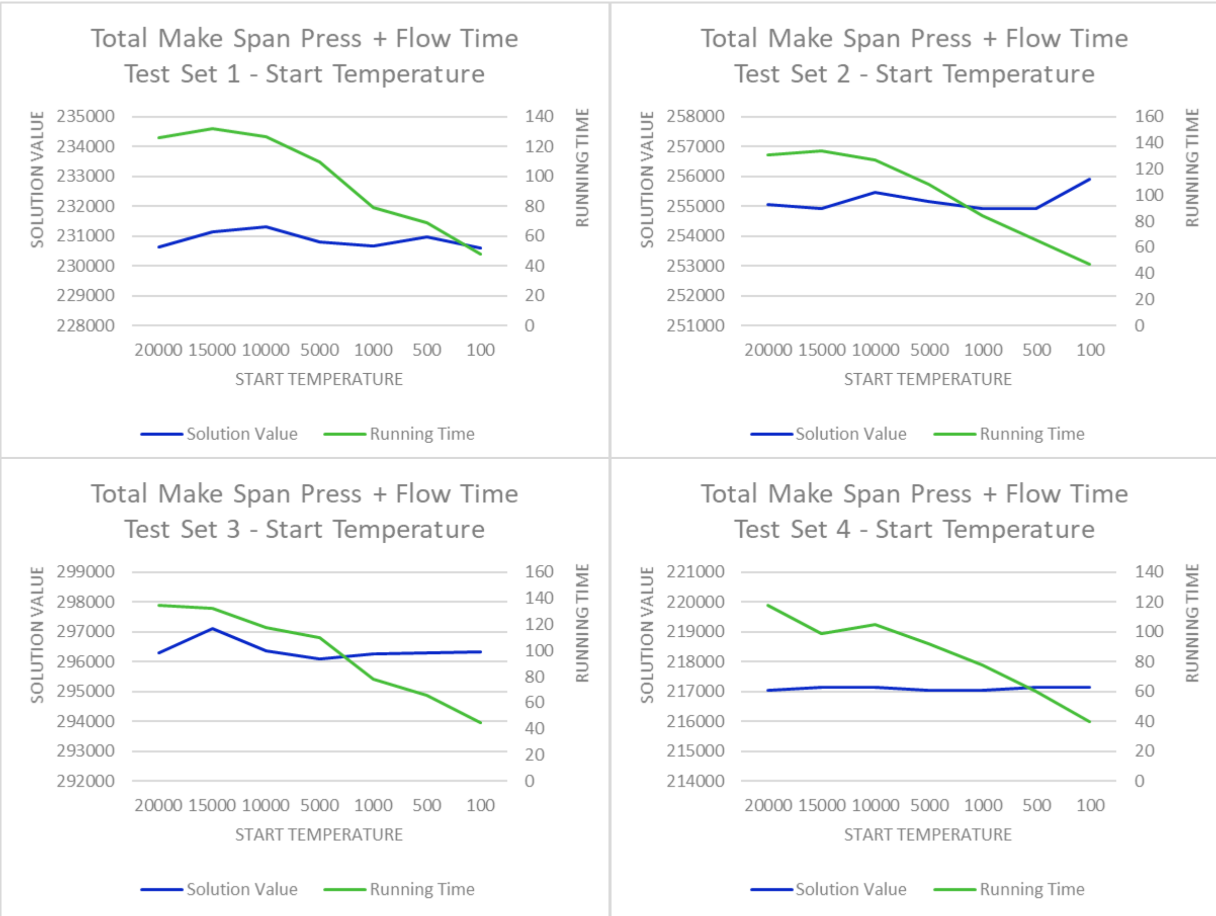

FIGURE 5.8: SENSITIVITY ANALYSIS TOTAL MAKE SPAN PRESS + FLOW TIME OBJECTIVE START TEMPERATURE

Figure 5.9 shows the solution value and running time for different values of the decrease factor or markov chain length for the TMSPFT objective. Using a smaller markov chain length does not change the solution value that much. For markov chain lengths of 100 or 200, the objective value is still more or less the same. However, this takes only 60 seconds, a reduction of the running time of 50%. For the decrease factor, we see an increase in solution value when the decrease factor and running time decreases. An exception is test set 4, which also has good solution values for smaller decrease factors.

When the running time is reduced by 100 seconds, e.g. by using a decrease factor of 0.75, the solution value increases with more or less 1000 seconds (17 minutes). If we assume that the grind and mix line meets the intermediate press line due dates, then the total flow time at the grind and mix line is 17 minutes higher. This is a bit more than 5 minutes per machine.

Figure 5.8 and Figure 5.9 show that the simulated annealing parameters on their own are not very sensitive. Decreasing one of them a bit does not really change the solution value. The solution space for the grind and mix line is smaller, because the press schedule is fixed. Jobs of the third product at a press line cannot be produced at the grind and mix line before all jobs of the first product are produced. There are only 2 silos available; so, storage problems arise and the schedule is infeasible. Hence, only relatively small changes are allowed to stay within the precedence constraints.

Total Tardiness

When we reduce the due dates by 50% and look at the total tardiness, no zero-tardiness schedule is feasible anymore. For Set 2 to 6, the time each press line becomes available for the set of jobs differs a lot. Therefore, the grind and mix line is able to supply each press line in time. We saw in our preceding analysis that simulated annealing can easily find a schedule for this case, also with lower parameter settings. For Set 1, which starts with an empty system, the algorithm needs to decide in which order the press lines need to start, to minimize the total tardiness. Therefore, the sequence at the grind and mix line is more important now. Hence, we expect the simulated annealing parameters to be more sensitive. Figure 5.10 shows that for decrease factors below 0.95, the total tardiness starts to increase. For markov chain lengths below 100, the total tardiness also starts to increase. Hence, we need a higher decrease factor than for the TMSPFT objective with zero tardiness, discussed above. The markov chain length should also be slightly higher.

To optimize the total tardiness, the press schedule is very important. The best total tardiness after simulated annealing depends on the total tardiness we obtain by Adaptive Search when we optimize the press schedule. Therefore, we perform a sensitivity analysis on the Adaptive Search parameters, to determine their sensitivity regarding the total tardiness objective. Figure 5.11 shows that the more iterations we use, the smaller the total tardiness is that we find. By doing more iterations, the probability of finding the best solution increases.

For higher bias factors, the construction of a schedule becomes more deterministic. Therefore, more iterations does not necessarily result in a smaller total tardiness. For low bias factors, more randomness causes more different schedules. So, more iterations means that more different schedules can be evaluated. Hence, the larger the probability that we find the best one. Interesting to see is that in Set 3, the best total tardiness solution is found with bias factors of 7 or higher, already from 10,000 iterations. The due dates are less tight than for Set 2 and 6, which show larger total tardiness values. Therefore, the algorithm has less difficulty with finding the best solution. For Set 2, a bias factor of 25 gives the best solution values; for Set 6, a bias factor of 10 performs better. For Set 2, a bias factor of 10 performs just as well as a bias factor of 25, for 50,000 iterations or more.

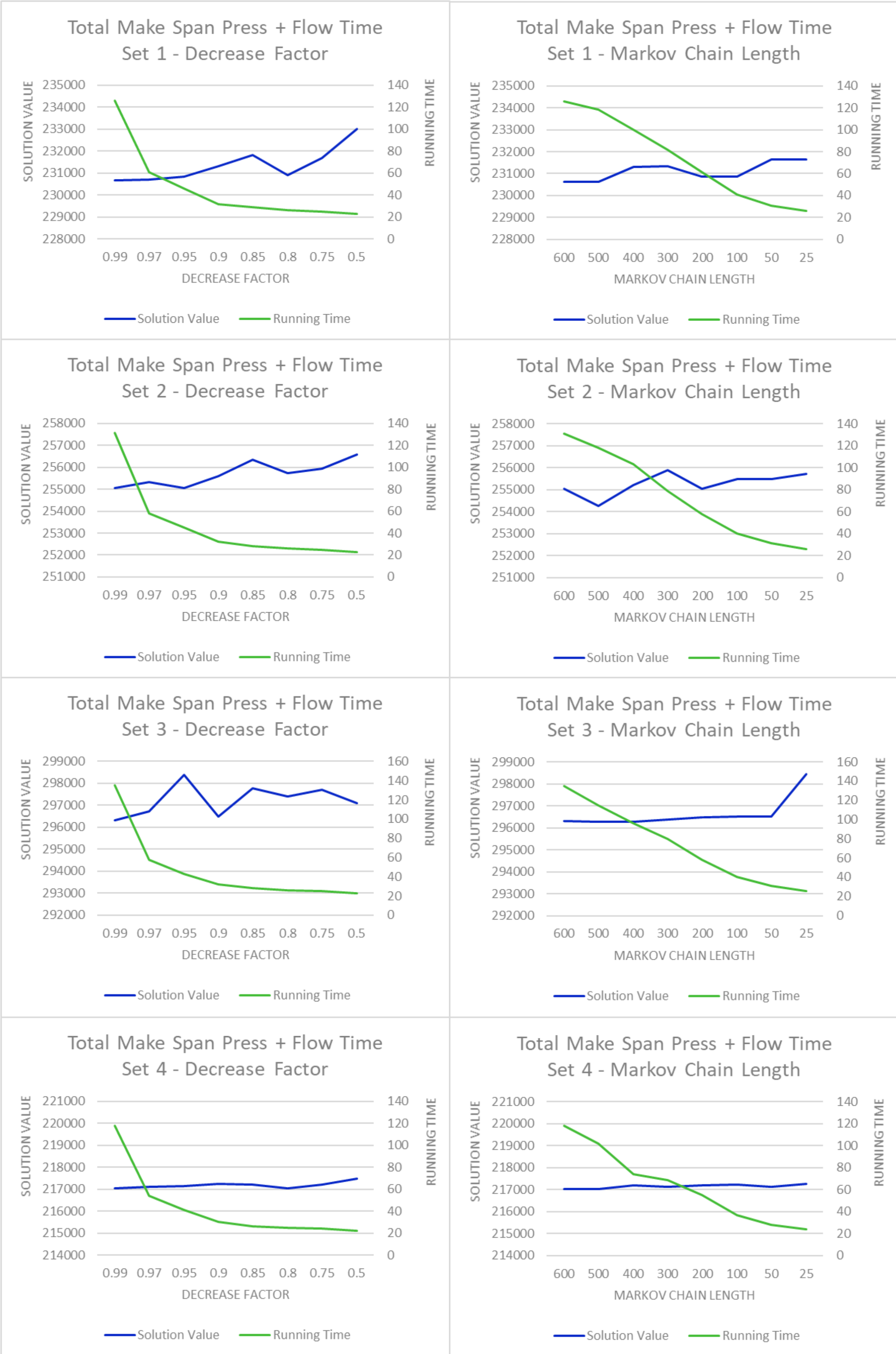

FIGURE 5.9: SENSITIVITY ANALYSIS TMSPFT OBJECTIVE

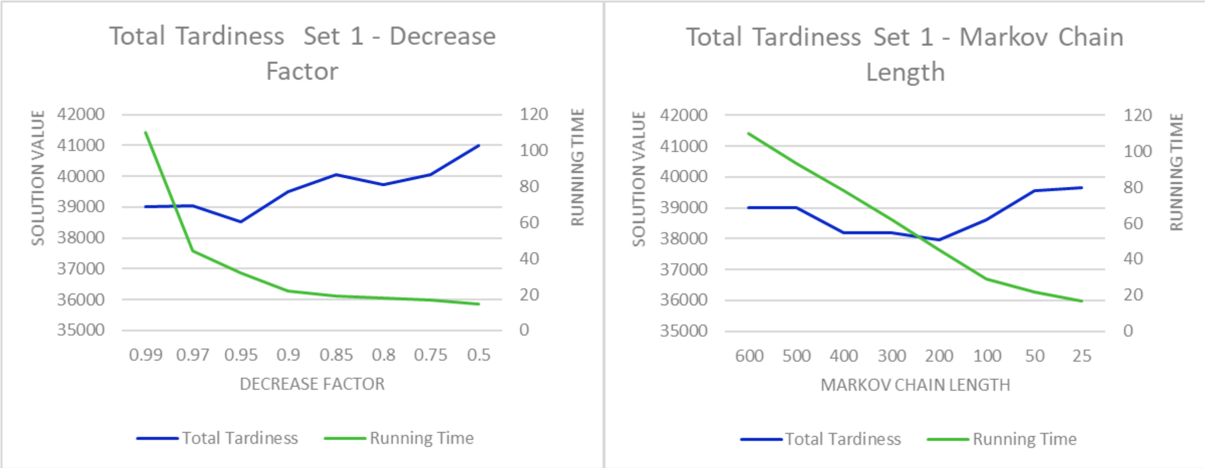

FIGURE 5.10: SENSITIVITY ANALYSIS TOTAL TARDINESS OBJECTIVE, SIMULATED ANNEALING PARAMETERS WITH REDUCED DUE DATE (50%)

FIGURE 5.11: SENSITIVITY ANALYSIS TOTAL TARDINESS OBJECTIVE, ADAPTIVE SEARCH PARAMETERS WITH REDUCED DUE DATE (50%)

5.3.4 Feasibility schedule

For the scope of this thesis, we made some decisions to be able to conduct our research within limited time. An important decision to limit the scope is to omit the transport to the finished product silo and the storage in finished product silos. In this section, we investigate whether this decision still results in feasible schedules, or that further research is needed. For storage in finished product silos, three categories are possible. (i) The first category is make-to-stock (MTS). There are always enough MTS silos for MTS products and the capacity is high enough, i.e. no more is produced than fits in the silo. (ii) The second category is make-to-order (MTO) pellets. There are 6 silos available for MTO pellet products. (iii) The third category is meal products. For meal products, only 2 silos are available. Therefore, meal products are also produced MTO.

Table 5.7 shows the number of silos that are required to store MTO pellets and meal products for the optimal schedules of Section 5.3.1, based on the make span and total make span press + flow time grind and mix line objectives. We use the time at which the job starts to be loaded into the truck, the due date, as the time the silo becomes available again. We assume the entire job is moved to small silos above the loading street (Section 1.1) at this moment in time. Next, we count the maximum number of products that needs to be stored at once.

Except for Set 6, all MTO pellet products can be stored in silos. Set 6 takes the largest period of time into account. Therefore, MTO pellet products can be produced much earlier before their due date. Thereby, they need to be stored for a longer period, which increases the number of silos needed. For meal products, only Set 1 is feasible. Set 1 consist of only 3 meal products. Hence, is more likely to be feasible with two silos. The other test sets require 4 to 11 meal silos. Recall that because there are only two meal silos, meal products are produced shortly before their due date. Therefore, an earliness objective for meal products would be interesting to investigate. However, it is an extra limitation and therefore it reduces the optimization potential. Modelling the storage in silos additionally to our model is another option. It is more challenging than trying an earliness objective but results in a schedule for which enough silos are available. Finally, ForFarmers could invest in more meal silos, or change the silo configuration such that more silos are dedicated for meal. A large range of meal products is produced in Set 2 to 6. More meal silos results in a more flexible production.

TABLE 5.7: NUMBER OF SILOS REQUIRED FOR OPTIMAL SCHEDULE

Set	Product Type	# Silos Make Span	# Silos Total Make Span Press + Flow Time
Set 1	MTO	3	4
	Meal	1	1
Set 2	MTO	4	4
	Meal	10	10
Set 3	MTO	3	4
	Meal	5	7
Set 4	MTO	3	3
	Meal	6	8
Set 5	MTO	5	6
	Meal	5	11
Set 6	MTO	7	7
	Meal	4	10

Besides that a silo needs to be available, the products also needs to be able to be transported to the silo by transport chains (Section 2.1). Each press line has its own primary chain and often needs a second chain (PK6, PK7, PK8, PK9, or MK2) to reach the silo. Appendix A shows these relations. Recall from Section 2.3 that not every silo can be reached by each transport chain. We manually check whether each product can be transported to its silo. Because of time restrictions, we were not able to model and to add the transport to the final silo to our model.

Figure 5.12 shows an assignment of chains to products produced at the press lines for the MS and TMSPFT objectives, for Set 1 and 5. Appendix E provides similar figures for the other test sets. Figure 5.12 shows each unique combination of products produced at the press lines per objective per test set, in chronological order. We start with a unique combination of 5 products at 5 press lines. When the last job of a product finishes, a job of another product starts and we get a new unique combination of 5 products. We name products by their product code. The number of combinations in which a product appears, does not say anything about the process time. By the grey colour, we indicate that a product is transported directly to the silo, without a second chain. The red colour indicates that there is no chain available. Hence, the schedule will be delayed at these lines. For five out of twelve schedules that we investigate, there is not a chain available for each product (see also Appendix E).

For the make span objective of test set 5, the problem would be solved when the 20175 is processed at line 5 and the products of line 5 at line 4. At line 5, the 20175 has a direct connection to its silo. However, this would have increased the make span. Nevertheless, PK7 would have been available for other products, solving the impossibilities in the schedule.

Directly	
PK6	
PK7	
PK8	
PK9	
MK2	
Not possible	

Set 1 — **MaxMakeSpan**

1	96015	96015	96414	97569	97569	97569	97569	97569	20235	20235	20235	96801
2		96455	96455	96455	20216	20216	20216	20216	20216	96405	96405	96405
3	96424	96233	96233	96233	96233	96233	96233	96432	96432	96432	96432	96432
4	96412	96412	96412	96412	96412	96412	96413	96413	96413	96413	96420	96420
5	96025	96025	96406	96406	96406	96421	96421	96421	96421	96421	96421	96421

FlowTime

1	96015	96015	96015	97569	97569	97569	97569	20235	20235	20235	96801	96801	96801
2	96455	96455	96455	96455	96455	96405	96405	96405	96405	96405	96405	96413	96413
3	96424	96414	96432	96432	96432	96432	96432	96432	96432	20216	20216	20216	20216
4	96233	96233	96233	96233	96233	96233	96412	96412	96412	96412	96412	96412	96412
5	96025	96025	96025	96025	96406	96406	96406	96406	96420	96420	96420	96420	96421

Set 5 — **MaxMakeSpan**

1	20436	20850	96446	96423	96423	96778	96403	96403	96403	96403	96403	96403	96411
2		96412	96412	96412	96412	96412	96412	96412	96406	96406	96406	96434	96434
3	96452	96452	96452	96452	96025	96025	96025	96432	96432	96432	96440	96440	96440
4		20175	20175	20175	20175	20175	20175	20175	20175	20175	20175	20175	20175
5					96404	96404	96404	96404	96404	20185	20185	20185	20185

FlowTime

1	96423	96423	96778	96778	96778	96778	96406	96406	20185	20185	20185	20185	20185
2	20850	20850	20850	20850	96403	96403	96403	96025	96025	96025	96025	96411	96411
3	96432	96446	96446	20436	20436	96452	96452	96452	96452	96452	96440	96440	96434
4			96412	96412	96412	96412	96412	96412	96412	96404	96404	96404	96404
5						20175	20175	20175	20175	20175	20175	20175	20175

FIGURE 5.12: TRANSPORT TO FINAL SILO TEST SET 1 AND 5

Notice that Set 1 does not use PK9. PK9 only has a connection with press line 1 (see Appendix A). For Set 1, only products that can be directly transported to their silo are produced at press line 1. For the other test sets, products at press line 1 that cannot be transported directly are transported by PK9. To increase the probability for the other lines to have a second chain available, the first press line should only produce MTO products (direct connection) or products that can be transported by PK9.

The silo of the 96413 in the optimal make span schedule and of the 96406 in the optimal total make span press + flow time schedule can both be reached by the PK9 chain. When these lines would have a connection to this chain, the infeasibility problem would be solved.

5.4 CONCLUSION

In this chapter we determined the performance improvement and we provided some insights for ForFarmers. By using our scheduling algorithm, we find schedules with zero tardiness except for one test set. Given the optimal tardiness value, we minimize three different make span objectives. First, the make span (MS), which is equal to the time the last job finishes. The make span can be improved by on average 8.2%. Second, the total make span (TMS), which is equal to the sum of the times that the last job finishes per machine. The total make span is improved by on average 2.6%. The last objective is the sum of the make span per press line, plus the flow time per job per machine of the grind and mix line (TMSPFT). The last objective can be improved by 4.8%.

Interesting to see is that the total make span objectives force each press line to start as soon as possible. This increases the utilization and because the press lines are the bottleneck for ForFarmers in Deventer, they need to run as much of the available time as possible. The make span objective does not force each line to start as soon as possible. The lines that are not responsible for the make span, can start at any time, in so far they finish before the line that determines the make span.

If we look in more detail to the grind and mix line only, we conclude that it is optimal to cluster as much jobs of the same product as possible. By clustering jobs of the same product, we save setup times in stage 3, which improves the make span or flow time objective. If we remove the setup time, e.g. because we place a second elevator that can be used, the optimal sequence is not clustered per product anymore.

By changing the parameters of the algorithm, a solution of the same quality can be found in less time. Most promising is a reduction of the markov chain length to 100, which reduces the running time by 50%. Reducing the start temperature to 5,000, saves also 20 seconds. When the due dates are more tight, a bias factor of 10 provides good results. The tighter the due dates and the more jobs in the set, the more iterations are needed to find the best schedule. 50,000 to 100,000 iterations seem to be enough to find a near to optimal schedule.

Finally, we tested the feasibility of the optimal schedules for the (transport to) finished product silos. Except for test set 6, all MTO pellets can be stored. For meal products, we need at most 11 silos instead of two, which are available. A possible solution would be an earliness objective for meal products or to add the silos to our model such that availability is considered. For transport to the finished product silo, often a second chain is needed of which only five are available. For 5 out of 12 optimal schedules we evaluated, there is no chain available for part of the production. This causes delay in the schedule because the press cannot continue if there is no chain to transport product to their silo.

6 Conclusion and Recommendations

This chapter provides an answer to the research problems of this thesis and gives recommendations and suggestions for further research. Section 6.1 draws conclusions to answer the research problems. Section 6.2 gives recommendations for ForFarmers to improve performance and suggestions for further research.

6.1 Conclusion

The first research problem we want to answer by our research is:

How can ForFarmers Deventer improve their operational production planning by optimizing the order sequence such that tardiness is minimized?

To improve the operational production planning by optimizing the order sequence we designed our own scheduling algorithm. The hybrid flow shop (HFS) problem is proven to be NP-hard. Thereby, no exact solution can be found for realistic problem sizes within reasonable time. Therefore, the algorithm that we propose is based on heuristics. We use a multi-objective optimization that consists of two parts. First, we optimize the maximum tardiness, total tardiness, or the number of jobs that is late. Given the best tardiness solution value we find, we optimize the make span, total make span of all machines, or the total make span of the press lines plus the flow time per job per machine of the grind and mix line.

We model the production process in five sequential stages. Each stage consist of one machine or a group of machines. The first stage consist of the sieve, hammer mill, and succeeding buffer 6. The second stage consist of the first mixer and the third stage of the buffer 7 combined with the second mixer. Stage four represents the press silos and connects the grind and mix line to the press lines. Stage five consist of the press lines.

The problem we have to solve is quite complex. Most stages not only have a processing time, but also an unloading time in which the product flows out of the machine. There is limited or no buffer capacity between stages. Therefore, unloading can only start when the next stage is available. Unloading also can only start when the current stage finishes processing, or a certain amount of time before it finishes. At the fifth stage, the press, unloading is defined as the time it takes to empty the cooling machine. The cooling machine only needs to unload between jobs of a different product. Therefore, jobs of the same product are often pelletized in a large batch. To combine jobs of the same product in one large batch, each press line has two preceding press silos. Each silo can store 30,000 kg. When a job finishes at the grind and mix line and it needs to be pelletized, it needs to be stored in one press silo. If there is no press silo available, or the destination press silo is full, the job blocks the grind and mix line and other jobs need to wait before they can start production. Therefore, jobs often start if the operator knows there is a press silo available. In practice, both press silos can be used to store jobs of the same product. It is quite challenging to implement this in our algorithm. Therefore, we assume that only one press silo can be used for jobs of the same product.

Processing and unloading times per product per machine are not exactly known. Therefore, we performed a statistical analysis to determine the process times based on their start date and finish date at each machine. These data is available in the MES. From this analysis we conclude that the process time of the hammer mill stage and buffer 7/ mixer 2 stage depend on the quantity of the job. No relation with the product is found. At the first mixer, the process time depends on the mixing program, which is product dependent. At the fifth stage, the press stage, the process time depends on the capacity at which the line is set. Capacities differ per product per line. Hence, the process time depend on the quantity, product and line at which the product is produced. The unload time at stage 1 depends on the quantity of the job. At stage 2 and 5, there is a fixed unloading time. For stage 5, this time differs per press line. Stage 3 and 4 do not have an unloading time. Stage 4 also does not have a processing time but is only used for intermediate storage.

In our algorithm, we use a divide-and-conquer strategy that divides the problem in two subproblems. In the first subproblem, we optimize the press lines with use of adaptive search. Adaptive search schedules job-by-job and chooses each time a job with a certain probability. We give a job that needs to finish first a higher probability to be chosen. We added an additional rule to the standard adaptive search algorithm that schedules all jobs of the same product after the first job of the product when it is chosen by adaptive search. For each job, when we schedule it at the press, we immediately assign it to a press silo. Thereby, we determine a release date and due date for the grind and mix line, such that the job is produced in time and fits in the press silo when it finishes processing at the grind and mix line. Next, we solve the second subproblem with use of simulated annealing. Given the optimal schedule at the press lines, we try to find a sequence for the grind and mix line that optimizes the overall objective value. Therefore, for each grind and mix line sequence we evaluate, we have to change the press schedule such that no job starts at the press before it finishes at the grind and mix line. When the schedule is feasible, we evaluate the objective value, according to the simulated annealing procedure. Not always a feasible solution is found, e.g. when we need more than two press silos because of a swap operation. We do not accept infeasible solutions as the current solution in our simulated annealing algorithm.

To test our algorithm we defined six test sets, representing different periods of time during the week. We programmed the algorithm in Embarcadero Delphi and we built a tool that can be used by ForFarmers to make scheduling decisions. Except for one test set, we found for each test set a solution with zero tardiness. For the test set with tardiness, this is because in this particular case our process time estimations for some jobs at the press line are larger than in reality. Thereby, the due date could not be met for these jobs. For the due date we use the time the job was loaded into the truck because we cannot trace back the original, official due date.

We analysed the three make span objectives to determine the performance improvement of our algorithm. For the make span objective, we found an average improvement of 8.2%, for the total make span objective an average improvement of 2.6%, and for the total make span press lines plus flow time grind and mix line an average improvement of 4.8%. The total make span improvement is lower because for four test sets, the make span at the grind and mix line increases. We have in total still a decrease of the total make span, because the make span at the press lines decreases more than that the make span at the grind and mix line stages increases. The increase in make span of the machines of the grind and mix line can partially be explained by our assumption that only one press silo can be used for jobs of one product.

All in all, we conclude that ForFarmers can improve the production planning by using an algorithm to optimize the sequence of jobs for production.

The second research problem we want to answer by our research is:

What are the bottlenecks in the process, before and after optimizing the production planning?

Section 2.5 showed that the press lines are the bottleneck for ForFarmers in Deventer. One grind and mix line can supply jobs to five press lines and to produce the required meal products. We determined the utilization for four weeks in which no large maintenance operations took place. The utilization of the grind and mix line in these weeks is 74% to 79%, based on the time it is processing or unloading the product. If we take the time the grind and mix line is occupied, sometimes products also have to wait until the next stage is available, the utilization is 83% to 91%. The utilization of the press lines, based on their actual process time, is 83% to 97%. Recall that the cooling machine needs to unload between jobs of different products. The utilization of the press lines, based on the time they are occupied, is 93% to 99.7%. Hence, the utilization of the press lines is always higher than the utilization of the grind and mix line.

The utilization of the grind and mix line is based on the hammer mill stage. The hammer mill stage is the bottleneck of the grind and mix line. It has the highest process and unloading time. Still, the fraction of time it is occupied is higher than the time it processes or unloads jobs. Hence, there are waiting times for succeeding stages. By scheduling production by our algorithm, these waiting times can be reduced (Table 5.6). In Section 5.3.2, we have seen that jobs of the same product are clustered at the grind and mix line to save setup times in stage 3. If jobs of the same product would not be clustered, there would be more setup times. In this case, the bottleneck shifts to stage 3, the buffer 7/ second mixer stage. If stage 3 is the bottleneck, the preceding stages have to wait more.

Optimizing the production planning does not change the bottleneck. Figure 5.4 shows gaps in the schedule for the grind and mix line, where all jobs at the press lines are connected to each other. Therefore, we know that the press lines are still the bottleneck. Table 5.4 shows that we save on average 5.1% on the total make span of the press lines. Hence, they are used more efficiently.

6.2 RECOMMENDATIONS AND FURTHER RESEARCH

This section provides recommendations and suggestions for further research for ForFarmers.

We recommend ForFarmers to schedule their production with use of algorithms. The first analyses in this thesis show promising results. Advantage is that the algorithm oversees a larger period compared to the operator, which makes decisions on intuition. Furthermore, the average age of operators is quite high and within 10 years, a lot of knowledge leaves the company. On the other hand, planning becomes more complex by more and more special products. Therefore, a scheduling algorithm can advise operators to make the right decision. The tool we programmed and used for our analysis has its limitations for daily use. Therefore, we recommend ForFarmers to find a software developer that can built a professional tool, which supports operators and is user friendly.

In general, we advise ForFarmers to batch jobs of the same product at the press lines as much as possible. It saves time because the cooling machines must empty less often. Thereby, more can be processed at the press lines, which are the bottleneck. At the grind and mix line, we advise to also cluster jobs of the same product. It saves ‘nalooptijd’ of the elevator transporting system after the second mixer. The ‘nalooptijd’ also creates waiting time at the preceding stages.

We assume that there is always a finished product silo, and a transport chain to reach the silo, available. Section 5.3.4 shows that this is not always the case. For 5 out of 12 best schedules, not every job can reach its silo by a transport chain. Therefore, we recommend to further analyse these resources and to add them to the algorithm. A possible way to include them in the algorithm is to add them as a restriction when choosing a press line. In our algorithm, often the earliest available line is chosen. A check needs to be added whether the destination silo can be reached by this press line via one of the chains that is available. If not, the job needs to be scheduled when a chain that is able to reach the silo becomes available, at the same or at another press line.

In Section 5.3.4, we discussed that chain PK9 could only be used in connection with the first press line. However, this line often processes jobs that can be transported to a silo without using a second chain. Thereby, the PK9 is unused. We advise ForFarmers to investigate whether PK9 can be connected to more press lines. This would increase flexibility when scheduling production.

We also saw in Section 5.3.4 that for make-to-order (MTO) pellet products, there is almost always a silo available. Hence, the total of six MTO pellet silos is sufficient. For meal products, the total of 2 silos is insufficient. We need at most 11 meal silos for one of our optimal schedules. We recommend three possible ways to deal with the limited number of silos:

1. Add them to the algorithm, such that the schedule takes the availability of MTO or meal silos into account. Jobs are not produced if no silo is available.
2. A different objective function that optimizes, besides tardiness, the earliness of meal jobs. Thereby, meal jobs are finished short before their due date, which is how operators schedule them currently.
3. Increase the amount of meal silos, e.g. by building new ones. Obviously, this is not the cheapest option. However, more meal silos combined with option 1 or 2 increases flexibility in the production schedule for the grind and mix line. Hence, it makes more improvements possible.

For the plant in Deventer, we found sufficient ground to exclude the dosing and weighing systems from our model. However, investments or changes in receipts can change this situation. Therefore, further research is needed on how to add the dosing and weighing systems to the algorithm. Recall from Section 2.4 that it is complex to determine process times for the dosing and weighing systems, because they depend on the quantity, product, and availability of ingredients and their distribution over dosing and weighing systems. The availability of ingredients and their distribution over dosing and weighing systems changes over time. Furthermore, multiple dosing and weighing systems can work at the same job at the same time.

When ForFarmers wants to implement the algorithm in another plant, a new data analysis should take place on the process data of that plant to determine its setup, process, and removal times. Furthermore, relations between stages needs to be assessed, e.g. when a certain stage can start after the previous stage. For another plant, it can be necessary to have the dosing and weighing systems added to the algorithm, because for some products succeeding stages needs to wait for it. ForFarmers obtains a manual of the tool we created, which explains more about the required data.

We assume that only one press silo can be used to store jobs of the same product above a press silo. We explained that in practice both silos can be used, but we use this assumption to reduce complexity and because it only has limited influence at the grind and mix line schedule. Further research can be done to test this assumption and to determine how a control rule can be implemented that can use both silos for jobs of the same product.

We only analysed one solution approach based on the adaptive search and simulated annealing heuristic. Further analysis can be done to test different heuristics, comparing their solution values and running times. Even though we doubt whether a simulated annealing approach works at the press lines for the total make span objective (Section 4.4), it would be interesting to analyse this approach and to compare it to our algorithm. Another interesting approach can be to use steepest decent at the grind and mix line instead of simulated annealing. Steepest decent only accept neighbours that give a better solution value. Therefore, it results in a local optimum. The running time, solution value, and stability (do we find each time more or less the same value?) of this local optimum can be compared to the simulated annealing approach we use.

For Adaptive Search, we use one value for the bias factor when running our algorithm. A more advanced approach is to use a bias factor scheme. This scheme starts with a high value in the first few iterations, such that a deterministic solution is found. Next, in one or a couple of steps, the bias factor is lowered and iterations are performed with a lower bias factor that causes more random schedules. When building these schedules, they can be compared to the best schedule found and if they are already worse, the schedule will not be finished. This saves time by not putting more effort than needed in schedules that are far from optimal. More research is needed to investigate how such a scheme would look like for ForFarmers in Deventer, whether it saves much running time and whether better solution values are found.

In Section 5.3.2 we already get an idea what happens if the grind and mix line would be the bottleneck. Further research can show the performance of our algorithm in case the grind and mix line is the bottleneck. We expect that there will be gaps in the press schedule, because the grind and mix line is not able to supply each line in time. Further research is needed to confirm this.

Because due dates are not really a problem for our test sets, we schedule all jobs of the same product after each other, just as operators do in practice. When due dates are more tight and no zero-tardiness schedule can be found, it can be interesting to allow jobs of the same product to be produced in multiple smaller batches. Further research is needed to test such a rule and whether it improves the tardiness objective for tighter due dates.

We performed a sensitivity analysis for due dates reduced by 50% and the influence of adaptive search parameters on the total tardiness. More research can be done to further investigate the performance for tighter due dates. For example, by taking new due dates at 25% or 75% of the current value. The sensitivity of these due dates can be compared with our analysis.

REFERENCES

Aarts, E., & Lenstra, J. K. (1997). *Local Search in Combinatorial Optimization.* John Wiley and Sons Ltd.

Acero-Domfnquez, M. J., & Patermina-Arboleda, C. D. (2004). Scheduling jobs on a k-stage flexible flow shop using a toc-based (bottleneck) procedure. In M. H. Jones, S. D. Patek, & B. E. Tawney, *Proceedings of the 2004 Systems and Information Engineering Design Symposium* (pp. 295-298). IEEE Press.

Adams, J., Balas, E., & Zawack, D. (1988). The Shifting Bottleneck procedure for job shop scheduling. *Management Science 34*, 391-401.

Alibaba. (2018, june 23). *Flexible Chain Conveyor for Feed Mill Manufacturer [Photograph]*. Retrieved from https://www.alibaba.com/product-detail/flexible-Chain-Conveyor-for-feed-mill_60612089721.html

Almex Extrusion Techniques. (2018, june 23). *BOA Compactor [Photograph]*. Retrieved from https://www.almex.nl/nl/producten/ancillary-equipment-2-nl/boa-compactor/

Artica. (2018, june 23). *Brand ForFarmers [Illustration]*. Retrieved from https://artica.nl/nl/werk/forfarmers

Arvos-Group. (2018, june 23). *Bucket Elevator Systems [Illustration]*. Retrieved from http://www.arvos-group.com/divisions/raymond-bartlett-snow/products/ehrsam-products/bucket-elevator-system/

Baker, K. R., & Bertrand, J. W. (1982). A Dynamic Priority Rule for Scheduling Against Due-Dates. *Journal of Operations Management*, 37-42.

Botta-Genoulaz, V. (2000). Hybrid flow shop scheduling with precedence constraints and time lags to minimize maximum lateness. *International Journal of Production Economics 64 (1-3)*, 101-111.

Brah, S. A., & Hunsucker, J. L. (1991). Branch and bound algorithm for the flow-shop with multiple processors. *European Journal of Operational Research 51 (1)*, 88-99.

Brah, S. A., & Loo, L. L. (1999). Heuristics for scheduling in a flow shop with multiple processors. *European Journal of Operations Research 113 (1)*, 113-122.

Brah, S. A., & Wheeler, G. E. (1998). Comparison of scheduling rules in a flow shop with multiple processors: a simulation. *Simulation 71 (5)*, 302-311.

Carlier, J., & NTron, E. (2000). An exact method for solving the multi-processor flow-shop. *RAIRO Recherche OpTrationelle 34 (1)*, 1-25.

Chen, C. L. (2009). Bottleneck-based heuristics to minimize total tardiness for the flexible flow line with unrelated parallel machines. *Computers and Industrial Engineering 56 (4)*, 1393-1401.

Cheng, C. L. (2008). Bottleneck-based heuristics to minimize tardy jobs in a flexible flow line with unrelated parallel machines. *International Journal of Production Research 46 (22)*, 6415-6430.

Cheng, J. L., Karuno, Y., & Kise, H. (2001). A shifting bottleneck approach for a parallel machine flowshop scheduling problem. *Journal of the Operations Research Society of Japan 44 (2)*, 140-156.

Dessouky, M. M., Dessouky, M. I., & Verma, S. K. (1998). Flowshop scheduling with identical jobs and uniform parallel machines. *European Journal of Operational Research 109 (3)*, 620-631.

Ding, F. Y., & Kittichartphayak, D. (1994). Heuristics for scheduling flexible flow lines. *Computers and Industrial Engineering 26 (1)*, 27-34.

DirectIndustry. (2018, june 23). *Animal Feed Production Expander [Illustration]*. Retrieved from http://www.directindustry.com/prod/amandus-kahl/product-21943-429919.html#product-item_457549

FEECO International . (2018, june 23). *Mechanical Construction of a hammer mill - 3D drawing by FEECO International [Illustration]*. Retrieved from http://feeco.com/hammer-mills/

Finke, D. A., Medeiros, D. J., & Traband, M. T. (2007). Multiple machine jit scheduling: a tabu search approach. *International Journal of Production Research 45 (21)*, 4899-4915.

Giebels, M. M. (2000). *EtoPlan: a concept for concurrent manufacturing planning and control.* Enschede: Universiteit Twente.

Gooding, W. B., Pekny, J. F., & Mccroskey, P. S. (1994). Enumerative approaches to parallel flowshop scheduling via problem transformation. *Computers and Chemical Engineering 18 (10)*, 909-927.

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.

Graves, S. C. (1981). A Review of Production Scheduling. *Operations Research 29*, 646-675.

Guinet, A. G., Solomon, M. M., Kedia, P. K., & Dussauchoy, A. (1996). A computational study of heuristics for two-stage flexible flowshops. *International Journal of Production Research 34 (5)*, 1399-1415.

Guirchoun, S., Martineau, P., & Billaut, J. C. (2005). Total completion time minimization in a computer system with a server and two parallel processors. *Journal of the Operational Research Research Society 32 (3)*, 599-611.

Gupta, J. N. (1988). Two-stage, hybrid flow shop scheduling problem. *Journal of the Operational Research Society 39 (4)*, 359-364.

Gupta, J. N., & Tunc, E. A. (1998). Minimizing tardy jobs in a two-stage hybrid flowshop. *International Journal of Production Research 36 (9)*, 2397-2417.

Gupta, J. N., Hariri, A. M., & Potts, C. N. (1997). Scheduling a two-stage hybrid flow shop with parallel machines at the first stage. *Annals of Operations Research 69*, 171-191.

Haouari, M., & M'Hallah, R. (1997). Heuristic algorithms for the two-stage hybrid flowshop problem. *Operations Research Letters 21 (1)*, 43-53.

Heerkens, H., & van Winden, A. (2012). *Geen Probleem.* Nieuwegein: Van Winden Communicatie.

Kim, Y. D., Joo, B. J., & Shin, J. H. (2009). Heuristics for a two-stage hybrid flowshop scheduling problem with ready times and a product-mix ratio constraint. *Journal of Heuristics 15 (1)*, 19-42.

Kolisch, R., & Drexl, A. (1996). Adaptive Search for Solving Hard Project Scheduling Problems. *Naval Research Logistics, Vol. 43*, 23-40.

Lee, G. C., & Kim, Y. D. (2004). A branch-and-bound algorithm for a two-stage hybrid flowshop scheduling problem minimizing total tardiness. *International Journal of Production Research 42 (22)*, 4731-4743.

Lee, Y. D., Kim, G. C., & Choi, S. W. (2004). Bottleneck focused scheduling for a hybrid flowshop. *International Journal of Production Research 42 (1)*, 165-181.

Lin, H., & Liao, C. (2003). A case study in a two-stage hybrid flow shop with setup time and dedicated machines. *International journal of production economics 86*, 133-143.

Lin, J. T., & Chen, C. (2015). Simulation optimization approach for hybrid flow shop scheduling problem in semiconductor back-end manufacturing. *Simulation Modelling Practice and Theory 51*, 100-114.

Liu, Y., & Karimi, I. A. (2008). Scheduling multistage batch plants with parallel units and no interstage storage. *Computers and Chemical Engineering 32 (4-5)*, 671-693.

Naderi, B., Zandieh, M., & Roshanaei, V. (2009). Scheduling hybrid flowshops with sequence dependent setup times to minimize makespan and maximum tardiness. *International Journal of Advanced Manufacturing Technology 41 (11-12)*, 1186-1198.

Phadnis, S., Brevick, J., & Irani, S. (2003). Development of a new heuristic for scheduling flow-shops with parallel machines by prioritizing bottleneck stages. *Journal of Integrated Design and Process Sciences 7 (1)*, 87-97.

Rajendran, C., & Chaudhuri, D. (1992). Scheduling in n-jobs m stage flowshop with parallel processors to minimize makespan. *International Journal of Production Economics 27 (2)*, 111-122.

Rao, T. B. (1970). Sequencing in the order a, b, with multiplicity of machines for a single operation. *OPSEARCH: Journal of the Operational Research Society of India 7*, 135-144.

Ruiz , R., & Maroto, C. (2006). A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. *European Journal of Operational Research 169 (3)*, 781-800.

Ruiz, R., & Vazquez-Rodríguez, J. A. (2010). The hybrid flow hop scheduling problem. *European Journal of Operation Research 205*, 1-18.

Salvador, M. S. (1973). A solution to a special case of flow shop scheduling problems. *S.E. Elmaghraby (Ed.), Symposium of the Theory of Scheduling and Applications*, 83-91.

Santos, D. L., Hunsucker, J. L., & Deal, D. E. (1996). An evaluation of sequencing heuristics in flow shops with multiple processors. *Computers and Industrial Engineering 30 (4)*, 681-691.

Sawik, T. J. (1993). A scheduling algorithm for flexible flow lines with limited intermediate buffers. *Applied Stochastic Models and Data Analysis 9 (2)*, 127-138.

Sawik, T. J. (1995). Scheduling flexible flow lines with no in-process buffers. *Internatinal Journal of Production Research 33 (5)*, 1357-1367.

Schutten, J. M. (1998). Practical job shop scheduling. *Annals of Operations Research 83*, 161-177.

Soman, C. A., van Donk, D. P., & Gaalman, G. (2004). Combined make-to-order and make-to-stock in a food production system. *International Journal of Production Economics 90*, 223-235.

Suresh, V. (1997). A note on scheduling of two-stage flow shop with multiple processors. *International Journal of Production Economics 49 (1)*, 77-82.

Thornton, H. W., & Hunsucker, J. L. (2004). A new heuristic for minimal makespan in flow shops with multiple processors and no intermediate storage. *European Journal of Operational Research 152 (1)*, 96-114.

Vairaktarakis, G., & Elhafsi, M. (2000). The use of flowlines to simplify routing complexity in two-stage flowshops. *IIE Transactions 32 (8)*, 687-699.

Varadarajan, A., & Sarin, S. C. (2006). A survey of dispatching rules for operational control in wafer fabrication. *IFAC Proceedings 39*, 715-726.

Vignier , A., Billaut, J. C., & Proust, C. (1999). Proust, Les problFmes d'ordonnancement de type flow-shop hybride: Ttat de l'art. *RAIRO Recherche OpTrationelle 33 (2)*, 117-183.

Voss, S. (1993). The two-stage hybrid flow shop with sequence-dependent setup times. In A. Jones, G. Fandel, & T. R. Gulledge , *Operations Research in Production Planning and Control, Proceedings of a Joint German/ US Conference* (pp. 215-220). Springer-Verlag.

Wang, X., & Tang, L. (2009). A tabu search algorithm for the hybrid flowshop scheduling with finite intermediate buffers . *Computers and Operations Research 36 (3)*, 907-918.

Wardono, B., & Fathi, Y. (2004). A tabu search algorithm for the multi-stage parallel machine problem with limited buffer capacities. *European Journal of Operational Research 155 (2)*, 380-401.

Wittrock, R. J. (1985). Scheduling algorithms for flexible flow lines. *IBM Journal of Research and Development 29 (4)*, 401-412.

Wittrock, R. J. (1988). An adaptable scheduling algorithm for flexible flow lines. *Operations Research 36 (3)*, 445-453.

Xiao, W., Hao, P., Zhang, S., & Xu, X. (2000). Hybrid flow shop scheduling using genetic algorithms. In *Proceedings of the Third World Congress on Intelligent Control and Automation* (pp. 537-541). IEEE Press.

Yang, T., Kuo, Y., & Chang, I. (2004). Tabu-search simulation optimization approach for flow-shop scheduling with multiple processors - a case study. *International journal of production research 42:19*, 4015-4030.

Yang, Y. (1998). *Optimization and Heuristic Algorithms for Flexible Flow Shop Scheduling, Ph.D. Thesis.* Columbia University.

Yaurima, V., Burtseva, L., & Tchernykh, A. (2009). Hybrid flowshop with unrelated machines, sequence-dependent setup time, availability constraints and limited buffers. *Computers and Industrial Engineering 56 (4)*, 1452-1463.

APPENDICES

APPENDIX A: RELATION STORAGE SILOS WITH PRESS LINES

APPENDIX B: PRESS LINES

APPENDIX C: TEST SETS

In this appendix, we present the time that each silo and press line becomes available per test set. Often our test set connects to jobs in production. Therefore, not all machines are available at time zero. The grind and mix line machines are all available at the start date, time zero. Press silos and press lines are often available at a later moment in time.

Test set 1

Start date: 18-02-2018 - 21:51:45.

TABLE C.1: SET 1 AVAILABLE FROM

Silo	Available from	Silo	Available from	Line	Available from
301	18-02-2018 - 21:51:45	302	18-02-2018 - 21:51:45	Press 1	18-02-2018 - 21:51:45
303	18-02-2018 - 21:51:45	304	18-02-2018 - 21:51:45	Press 2	18-02-2018 - 21:51:45
305	18-02-2018 - 21:51:45	306	18-02-2018 - 21:51:45	Press 3	18-02-2018 - 21:51:45
307	18-02-2018 - 21:51:45	308	18-02-2018 - 21:51:45	Press 4	18-02-2018 - 21:51:45
309	18-02-2018 - 21:51:45	310	18-02-2018 - 21:51:45	Press 5	18-02-2018 - 21:51:45

Test set 2 + 6

Start date: 25-01-2018 - 02:04:21.

TABLE C.2: SET 2+6 AVAILABLE FROM

Silo	Available from	Silo	Available from	Line	Available from
301	25-01-2018 - 03:04:00	302	25-01-2018 - 04:32:00	Press 1	25-01-2018 - 04:32:00
303	25-01-2018 - 04:26:00	304	25-01-2018 - 07:39:00	Press 2	25-01-2018 - 07:39:00
305	25-01-2018 - 02:20:00	306	25-01-2018 - 04:05:00	Press 3	25-01-2018 - 04:05:00
307	25-01-2018 - 02:58:00	308	25-01-2018 - 07:16:00	Press 4	25-01-2018 - 07:16:00
309	25-01-2018 - 02:04:21	310	25-01-2018 - 03:33:00	Press 5	25-01-2018 - 03:33:00

Test set 3

Start date: 26-01-2018 - 02:26:34.

TABLE C.3: SET 3 AVAILABLE FROM

Silo	Available from	Silo	Available from	Line	Available from
301	26-01-2018 - 02:47:00	302	26-01-2018 - 04:59:00	Press 1	26-01-2018 - 04:59:00
303	26-01-2018 - 05:40:00	304	26-01-2018 - 10:33:00	Press 2	26-01-2018 - 10:33:00
305	26-01-2018 - 03:50:00	306	26-01-2018 - 08:46:00	Press 3	26-01-2018 - 08:46:00
307	26-01-2018 - 02:26:34	308	26-01-2018 - 06:01:00	Press 4	26-01-2018 - 06:01:00
309	26-01-2018 - 02:31:00	310	26-01-2018 - 04:43:00	Press 5	26-01-2018 - 04:43:00

Test set 4

Start date: 03-02-2018 - 06:36:28.

TABLE C.4: SET 4 AVAILABLE FROM

Silo	Available from	Silo	Available from	Line	Available from
301	03-02-2018 - 06:36:28	302	03-02-2018 - 10:01:41	Press 1	03-02-2018 - 10:01:41
303	03-02-2018 - 10:08:41	304	03-02-2018 - 13:22:00	Press 2	03-02-2018 - 13:22:00
305	03-02-2018 - 06:36:28	306	03-02-2018 - 12:32:12	Press 3	03-02-2018 - 12:32:12
307	03-02-2018 - 06:36:28	308	03-02-2018 - 09:54:45	Press 4	03-02-2018 - 09:54:45
309	03-02-2018 - 06:36:28	310	03-02-2018 - 09:50:06	Press 5	03-02-2018 - 09:50:06

Test set 5

Start date: 09-01-2018 - 16:14:00.

TABLE C.5: SET 5 AVAILABLE FROM

Silo	Available from	Silo	Available from	Line	Available from
301	09-01-2018 - 16:14:00	302	09-01-2018 - 17:27:00	Press 1	09-01-2018 - 17:27:00
303	09-01-2018 - 16:14:00	304	09-01-2018 - 19:35:00	Press 2	09-01-2018 - 19:35:00
305	09-01-2018 - 16:14:00	306	09-01-2018 - 16:35:34	Press 3	09-01-2018 - 16:35:34
307	09-01-2018 - 16:14:00	308	09-01-2018 - 20:16:00	Press 4	09-01-2018 - 20:16:00
309	09-01-2018 - 16:14:00	310	09-01-2018 - 22:00:00	Press 5	09-01-2018 - 22:00:00

APPENDIX D: MODEL PARAMETERS

The optimization strategy we propose consist of several parameters. To perform experiments, we must determine appropriate values for these parameters. We want to build an online planning tool. Hence, the running time may not be too large. On the other hand, the solution value should be as good as possible. So, there is a trade-off between running time and solution value. This appendix explains how we determine the used parameter values.

The simulated annealing (SA) part uses a so-called cooling scheme, consisting of four parameters: a start temperature (c_0), a stop temperature (c_{stop}), a decrease factor (α), and a markov chain length (k). For the adaptive search part, we have to determine the value for beta (β) and the number of iterations. Recall Section 4.4.

First, we set the beta to 100 and the number of iterations to 1000. Thereby, the probability that we get each time the same schedule as the best schedule is high. We want each time the same schedule because it makes comparing the performance of the simulated annealing parameters possible. First, we determine the start temperature (c_0) and stop temperature (c_{stop}). Therefore, we use the acceptance ratio which is related to the temperature c ($\chi(c)$). The acceptance ratio at a certain temperature is calculated by the following formula:

$$Acceptance\ ratio: \chi(c) = \frac{Number\ of\ accepted\ transitions}{Number\ of\ proposed\ transitions}$$

Recall from Section 4.4 that if the new transition solution is worse than the current solution, we accept the solution with a certain probability, based on the current temperature. As the current temperature decreases, the probability of accepting a worse solution decreases and so the acceptance ratio. Figure D.1 shows the relation between the acceptance ratio and the temperature. For higher temperatures, the acceptance ratio approaches one. We do not want a big range of temperatures in our cooling scheme with an acceptance ratio close to zero. It takes running time and is more or less a random search for a solution. Therefore, we set the start temperature where the acceptance ratio really starts to decrease when the temperature decreases (see Figure D.1). We want to stop when we do not find any better solutions, when the acceptance ratio is close to zero. We can have a whole range of temperatures where the acceptance ratio approaches zero. We do not want to include this whole range of temperatures. So, we take the temperature from where the acceptance ratio is more or less zero, when the temperature decreases (see Figure D.1).

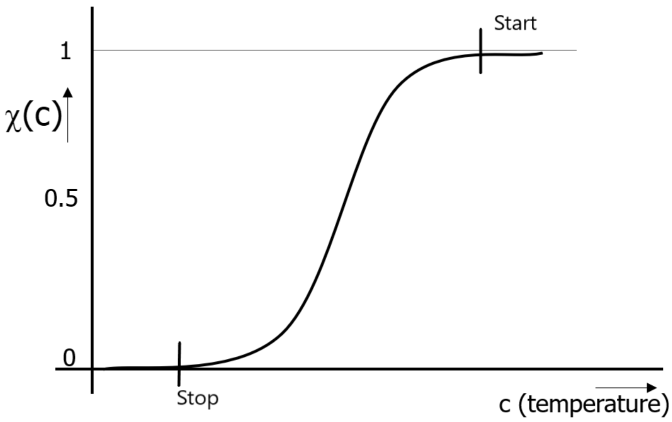

FIGURE D.1: RELATION ACCEPTANCE RATIO/ TEMPERATURE

Figure D.2 shows the acceptance ratio for high temperatures. We see that for temperatures below 10,000, the acceptance ratio really starts to decrease. Above 20,000, all ratios seem to be above 0.9. Between 10,000 and 20,000 there are already some points below 0.9. Therefore, we conclude that the acceptance ratio starts decreasing when the temperature drops below 20,000. Hence, the start temperature (c_0) becomes 20,000.

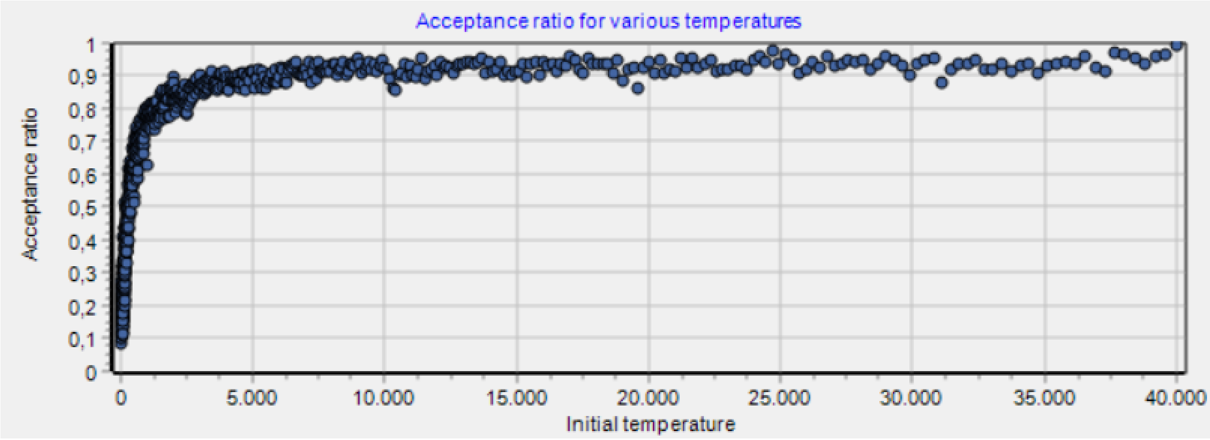

FIGURE D.2: ACCEPTANCE RATIO START TEMPERATURE

Figure D.3 shows the acceptance ratio for low temperatures, to determine the stop temperature (c_{stop}). We estimate that, based on Figure D.3, the stopping temperature is around 1.5. For the lowest temperatures, smaller than 1.5, all points seem to be at the x-axis.

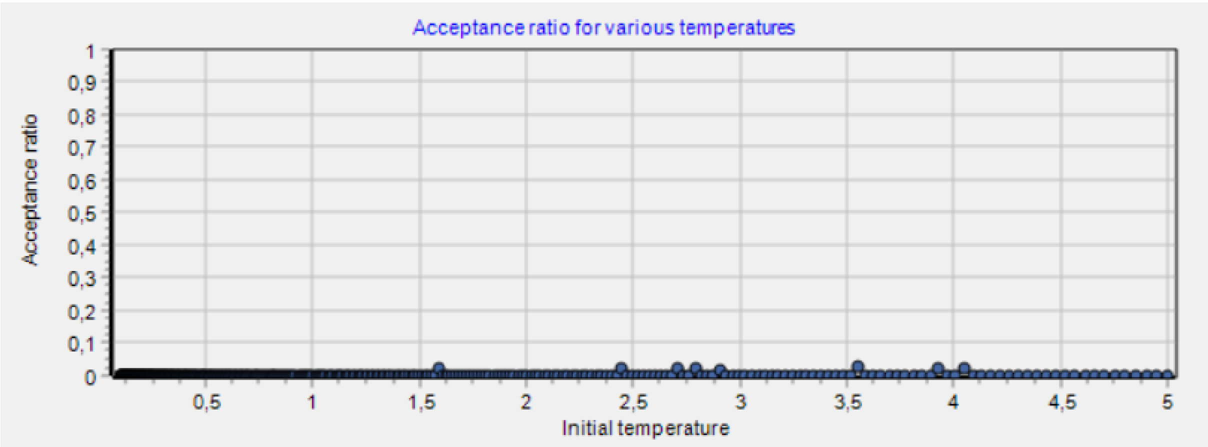

FIGURE D.3: ACCEPTANCE RATIO STOP TEMPERATURE

For the last two parameters of the SA cooling scheme, the decrease factor and markov chain length, we evaluate the solution value and running time for different decrease factors and markov chain lengths. We evaluate decrease factors: 0.95, 0.96, 0.97, 0.98, 0.99, and 0.995. For the markov chain length we evaluate lengths of: 100, 200, 400, 600, 800, and 1000.

Figure D.4 shows for multiple decrease factors the solution value per markov chain length. Figure D.5 shows for multiple markov chain lengths the solution value per decrease factor. Both figures show that for a high value of the markov chain length or decrease factor, the impact of the other parameter decreases. The variability in solution values decreases (the lines are closer to each other) for higher markov chain lengths or decrease factors. If we choose one to be low, then we need to choose a high value for the other parameter. Table D.1 shows the running times per decrease factor, per markov chain length. Of course, when the decrease factor or markov chain length increases, the running time increases.

Based on both figures, we decide to use a decrease factor of 0.99 and a markov chain length of 600. The corresponding running time is 122 seconds. Using a 'slower' cooling scheme does not give a better solution value and takes more running time. Our scheme gives one of the best solution values we found.

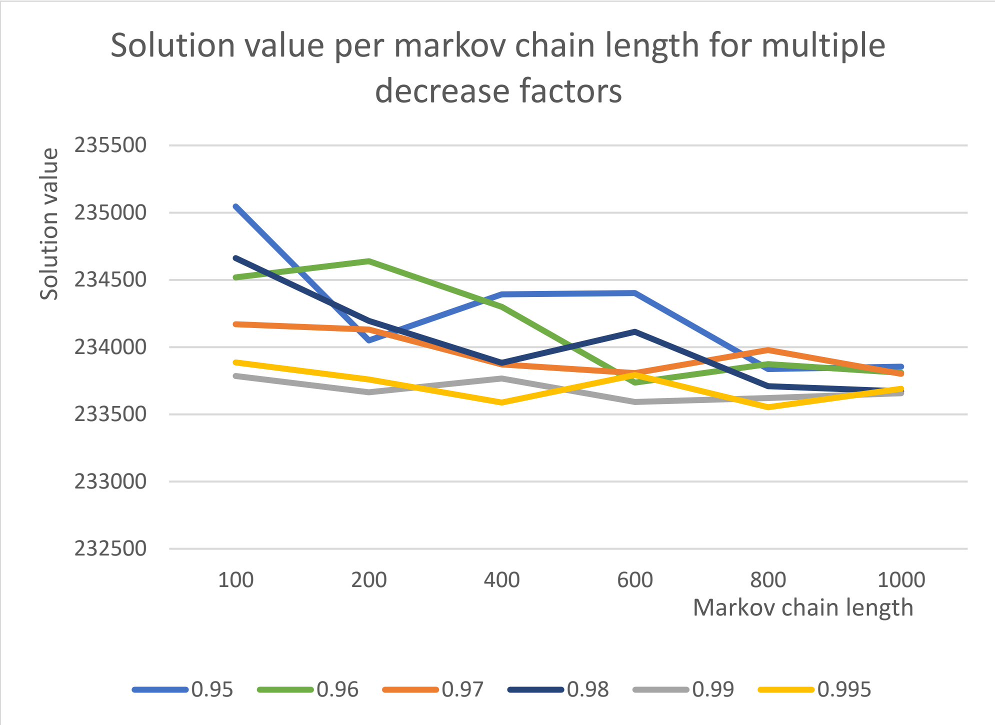

FIGURE D.4: SOLUTION VALUE PER MARKOV CHAIN LENGTH

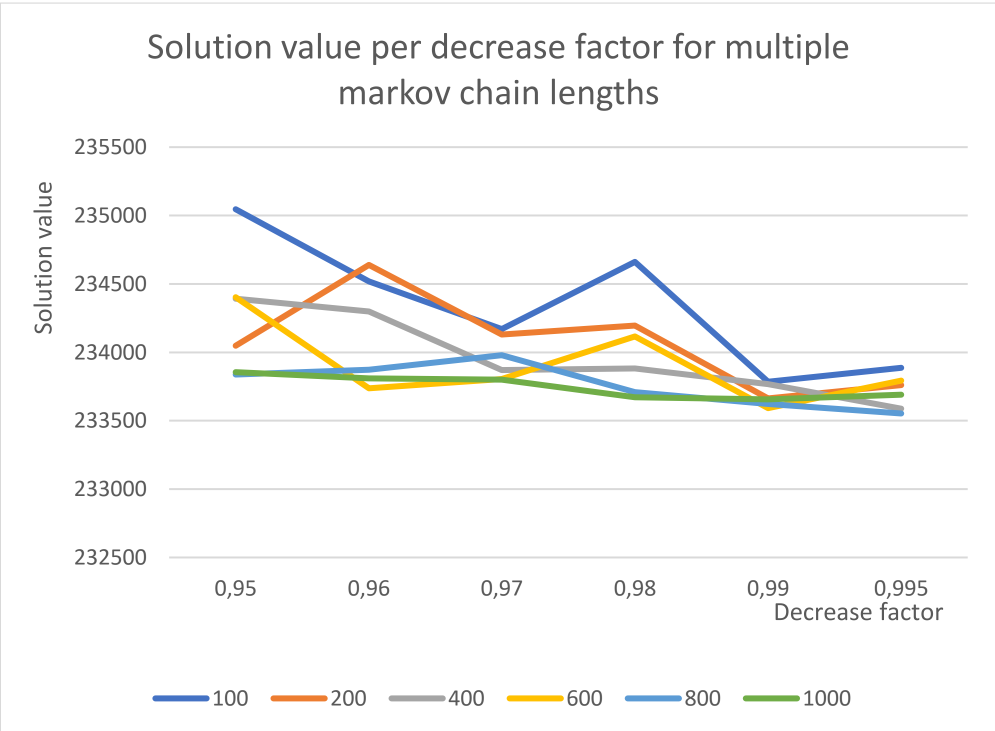

FIGURE D.5: SOLUTION VALUE PER DECREASE FACTOR

TABLE D.1: RUNNING TIME PER DECREASE FACTOR PER MARKOV CHAIN LENGTH (TIME IN SECONDS)

Markov chain Length	0.95	0.96	0.97	0.98	0.99	0.995
100	6	7	8	12	22	42
200	10	11	15	22	41	81
400	17	21	28	41	80	160
600	25	32	41	61	122	236
800	33	41	54	80	165	327
1000	40	51	66	99	197	402

To determine the value for the AS parameters we set the values for the SA parameters and decrease the AS parameters. First, we decrease the beta (β), the bias-factor. The higher the bias-factor, the more deterministic the construction of a schedule becomes. Hence, the probability of choosing the job with the highest regret factor is higher. For lower beta values, this probability is lower. The lower probability makes it possible to construct more different schedules which can be better in the end.

Table D.2 shows that for smaller beta values, in general better solutions are found. However, we need more iterations. The total make span of the press lines per parameter combination per test set in Table D.2, is an average of three replications. If we use a low beta value, 1000 iterations are not sufficient anymore. If we use one of these bias factors, 10,000 iterations are sufficient. Using 20,000 iterations does not guarantee that a better solution will be found. We decide to use 10,000 iterations and a bias-factor of 3. The low bias-factor increases the probability of finding different schedules. The higher total amount of iterations increases the probability of finding the optimal schedule. It does not seem to be useful to use more than 10,000 iterations for test set 1 to 4. It gives no further improvement. Using 10,000 iterations instead of 1000 iterations increases the running time by 11 to 20 seconds.

Test set 5 and 6 consist of more jobs. Using more iterations makes more sense here. More jobs means that there are more schedules possible. Therefore, it takes more iterations before a good schedule is found. Remarkable is that the best solutions for test set 6 are found for higher bias-factors. Set 6 is the largest test set, taking almost one day of production into account. With smaller bias-factors, a lot of schedules are created where jobs with a larger due date are in the first halve of the schedule (recall the probability of choosing a job, Section 4.4). Thereby, jobs that need to finish early in the schedule are produced too late. For low bias-factors no schedule with zero tardiness is found anymore, we indicate this with 'inf' in Table D.2. For higher bias-factors, a zero-tardiness schedule is found, which we prefer because tardiness is our first objective. By using higher bias-factors in the range 20 to 60, the weight of the regret factor, the due date, counts more and the probability of choosing a job that needs to be finished first increases.

Based on our analysis, we use as rule of thumb: the bias-factor is equal to the maximum between 3 and the number of jobs minus 150. For the number of iterations, we stick to 10,000.

TABLE D.2: PARAMETERS ADAPTIVE SEARCH AND THEIR SOLUTION VALUE

		Total MakeSpan Press Set:						
Nr Iterations	**Beta**	**1**	**2**	**3**	**4**	**5**	**6**	**Running Time**
1000	100	202020	229934	271102	199024	322421	433951	2
1000	60	202020	229675	270886	199023	322421	432132	2
1000	40	202020	229335	270994	199024	322125	432270	2
1000	30	200021	229269	270886	199052	321357	431750	2
1000	20	199729	229096	270289	199127	321760	431540	2
1000	10	199298	229585	272226	199332	321039	433938	2
1000	5	199297	230246	271895	199226	321112	444172	2
1000	3	199590	230729	271537	199271	321666	Inf	2
1000	2	199843	230719	272541	199204	320317	Inf	2
1000	1	200762	230978	276169	199213	322754	Inf	2
10000	5	199297	228969	269828	199024	319582	435642	13-22
10000	3	199297	229353	270048	199024	319711	442986	13-22
10000	2	199297	229459	269871	198986	319521	Inf	13-22
10000	1	199492	229960	271750	199061	320547	Inf	13-22
20000	5					318022	433360	24-44
20000	3	199297	229388	270229	199024	318610	440098	24-44
20000	2	199297	229536	269651	199061	318865	440213	24-44
20000	1	199298	229678	270690	198986	319607	Inf	24-44

Appendix E: Feasibility transport to final silo

Set 2 — **MaxMakeSpan**

1	20850	20850	20862	20862	96010	96430	96430	96430	96415
2			96490	96490	96490	96490	96490	96403	96403
3	96452	96452	96452	96452	96452	96496	96440	96440	96440
4		96412	96412	96408	96408	96408	96408	96408	96408
5	20175	96404	96404	96404	96404	96404	96404	96404	96404

FlowTime

1	96496	96496	20850	20850	20850	96430	96430	96430	20862	20862	96415
2				96403	96403	96403	96490	96490	96490	96490	96490
3	96452	96452	96452	96452	96452	96452	96452	96010	96010	96440	96440
4				96404	96404	96404	96404	96404	96404	96404	96404
5	96412	20175	20175	20175	96408	96408	96408	96408	96408	96408	96408

Set 3 — **MaxMakeSpan**

1	96513	96057	96423	96423	96411	96411	96411	96411	96411	96411
2						96415	96415	96415	96415	20216
3			20993	20993	20993	96293	96293	96405	96405	96405
4	96404	96404	96404	96420	96420	96420	96406	96406	96406	96406
5	20175	20175	20175	20175	20175	20175	20175	20175	96413	96413

FlowTime

1	96513	96423	96423	96057	96057	96415	96415	96415	96415	96405	96405
2							96411	96411	96411	96411	96411
3					20993	20993	96293	96293	96293	96293	20216
4	96420	96420	96420	96420	96420	96420	96404	96404	96413	96413	96413
5	96406	96406	20175	20175	20175	20175	20175	96421	96421	96421	96421

Set 4 — **MaxMakeSpan**

1	96436	20436	20436	20436	20436	96015	20778	20778	20778	96430
2						96439	96439	96439	96412	96412
3					96513	96513	96513	96513	96513	96513
4	20850	20175	20175	20175	20175	20175	20175	20175	20175	20175
5	96411	96411	96411	96411	96411	96411	96411	96025	96025	96025

FlowTime

1	96436	20436	96015	96015	96015	20197	20197	20778
2			96411	96411	96411	96411	96411	96411
3		96513	96513	96430	96430	96430	96439	96439
4	20850	96025	96025	96025	96412	96412	96412	
5	20175	20175	20175	20175	20175	20175	20175	

Set 6 **MaxMakeSpan**

1	20850	96010	96496	96521	96521	96665	96665	96665	20741	20741	96420	96420	96420	96440	96440	96440	96440	96440	96430
2	96415	96415	96415	96415	96415	96415	96415	96415	96415	96415	96415	96415	96415	96415	96415	96415	96490	96490	96490
3	96452	96452	96452	96452	20862	20862	20436	20436	20436	20495	20495	96025	96025	96025	96403	96403	96403	96403	96403
4	96412	96412	96412	96412	96412	96412	96412	96412	96412	96412	96412	96412	96401	96401	96401	96402	96402	96408	96408
5	96404	96404	96404	96404	96404	96404	96404	20175	20175	20175	20175	20175	20175	20175	20175	20175	20175	20175	20175

FlowTime

1	96496	20850	20850	20862	20862	96010	96010	96521	96521	96521	96521	96521	20741	20741	96430	96430	96430				
2			96415	96415	96420	96420	96420	96420	96420	96490	96490	96490	96490	96490	96490	96490	96490	96493	96493	96493	96493
3	96452	96452	96452	96452	96452	96452	96665	96665	20495	20495	20436	20436	20436	96440	96440	96440	96403	96403	96403		
4		96412	96412	96412	96412	96412	96412	96412	96412	96412	96412	96412	96412	96412	96412	96025	96025	96025	96402	96401	96401
5	96404	96404	96404	96404	96404	96404	96404	96404	96404	96404	96404	20175	20175	20175	20175	20175	20175	20175	20175	20175	96408