Developing a real-time job scheduling model for a packaging process with practical restrictions at Euroma

Using heuristics to optimize real-time scheduling of jobs with restrictions related to the food industry

07-09-2023

Loete Heijdenrijk
MSc Industrial Engineering and Management

UNIVERSITEIT TWENTE.

Developing a real-time job scheduling model for a packaging process with practical restrictions at Euroma

Author
Loete Heijdenrijk
MSc Industrial Engineering and Management

Company
Royal Euroma B.V.
Ravensburgstraat 4
8028 PZ Zwolle

University
University of Twente
Drienerlolaan 5
7522 NB Enschede

Supervisors university
Dr. ir. J.M.J. Schutten — *First supervisor*
Dr. ir. L.L.M. van der Wegen — *Second supervisor*

Supervisors company
T.S. van Benthem — *Unit Manager*
P. Bax — *Operations manager*

Date
07-09-2023

Preface

The past years have marked an unforgettable journey of learning and growth for me as I pursued my master's degree. During this time, I learned of subjects unknown to me, leading to new passions. Reflecting on these challenges, I find myself looking back with a sense of accomplishment, appreciating the invaluable experiences that have shaped me. I am now eager to apply the knowledge gained in the following chapters of my life.

I would like to take this opportunity to extend my gratitude to the persons who have been involved in my academic journey. First, I wish to express my appreciation to my university supervisors for their insightful guidance and support. Their mentorship enriched my understanding of the subject and helped shape my research.

I am also fortunate to have received unwavering encouragement from my family and friends throughout my study. I owe a special debt of gratitude to my sister Maaike, whose reading sessions during train rides refined my work, and Po and Luna for the perfect emotional support.

The opportunities provided by the company where I conducted my thesis research have been truly enriching. My heartfelt thanks go to my supervisors, Tim and Pim, for their exceptional guidance, continuous feedback, and willingness to share their knowledge and expertise. Additionally, I am grateful for the camaraderie and insightful advice I received from my colleagues at the office.

Lastly, I wish to extend my gratitude to my classmates, Fabian and Thomas. Our exchanges, discussions, and occasional venting provided a supportive network during this intense period, and I cherish the shared experiences.

While the path to this thesis was not always easy, I stand proud and content with the final outcome. It is my hope that this work not only reflects my dedication and efforts but also contributes meaningfully to the company. With genuine appreciation, I eagerly anticipate the opportunities that lie ahead as I continue to expand upon the knowledge and experiences gained during this journey.

With appreciation,

Loete Heijdenrijk

Management summary

Euroma, a company in spice-based product development and manufacturing, contains a mixing department that blends ingredients to create spice blends. As Euroma aims to increase production, a challenge emerged: the limited intermediate bulk container (IBC) availability in the mixing department. The current scheduling method for jobs in the packaging step causes delays in new job releases throughout the mixing department due to a backlog of filled IBCs.

The core goal of this study is to reduce the packaging process's lead time, improving IBC availability. The corresponding research question is:

How can the lead time at the packaging process be decreased to improve the availability of IBCs at Euroma?

The packaging process uses two machines, Votech and Dinnissen. Constraints related to colour changes, allergen-free product sanitation, and adhering to kosher and halal certifications restrict the sequencing on the machines. Additionally, the longevity of products within IBCs adds complexity.

An analysis of the literature and the situation at Euroma leads to formatting the problem as two single-machine scheduling problems. In addition, the literature study delves into solution methodologies, focusing on constructive and improvement heuristics. Next to adapting existing heuristics, we developed a heuristic that minimizes the number of dummy jobs needed to process certified jobs (MD).

We conducted experiments with the different heuristics. Adjusting the old schedule (AOS) and the NEH constructive heuristic, coupled with the SA improvement heuristic, yielded the lowest objective function values within a reasonable CPU time.

The results show an improvement in scheduling by the algorithms compared to the current situation. The reduction of the number of dummy jobs and cleanings needed caused a reduction from 17.0 hours of cleaning time on average per day to 11.5 (NEH) and 11.6 hours (AOS). However, the total amount of tardy jobs increased, but the number of extremely late jobs decreased.

An analysis of the number of IBCs necessary showcased using the model can decrease the average number of IBCs waiting for packaging from 23.5 in the current situation to 13.4. In addition, when experimenting with increased demand, the results showed a possible 20% demand increase before IBC use is at the current level.

NEH and AOS with SA both give a low use of dummy jobs. However, they do not limit the use, as MD does. Although MD may not be the best option for this problem, our algorithm can offer significant value when reducing dummy jobs is paramount. In addition, a potential improvement to the model includes incorporating a restriction for using the minimal number of dummy jobs, where the principles of MD can calculate this minimum.

The recommendations to Euroma include implementing the model, using NEH or AOS and SA. In addition, to addressing the challenge of tardy jobs, we advise exploring adjusting the weights used in the objective and introducing article-specific due dates. Furthermore, synchronizing mixer planning with machine operations can enhance efficiency.

To implement the solution in practise, the developed model needs to integrate into a practical tool, which extracts data from various systems and uses this to feed the model. Additionally, the tool translates the schedules generated by the model for use within the systems. The tool responds to specific events, such as machine vacancies and new job arrivals in queues. After creating a new schedule, the tool initiates a series of events, including schedule updates and activation of AGVs to move IBCs.

Technical adjustments to the current systems need to accommodate the tool. After the technical implementation of the tool, we propose testing in the factory and gathering feedback from operators and planners. Weekly evaluations guide ongoing adjustments. If needed, this phase extends to ensure a seamless integration. The last step is complete integration, where operators strictly follow the schedule, and the tool triggers AGVs to transport IBCs.

In conclusion, this thesis explores enhancing IBC availability by decreasing lead time in Euroma's packaging process. The findings, recommendations, and strategies offer the potential to boost operational efficiency in line with Euroma's production goals. However, refinement, practical trials, and integration are needed to realize these benefits.

Contents

List of figures

List of tables

1 Introduction

This thesis forms part of the graduation assignment for the master's program in Industrial Engineering and Management at the University of Twente. The research occurs at Royal Euroma B.V, which we refer to as Euroma.

Section 1.1 introduces Euroma, addressing the company and the department where the research takes place. Next, Section 1.2 shows the problem context and causes. Finally, Section 1.3 addresses the research objective, scope, research questions, deliverables, and the outline of the report.

1.1 Introduction to Euroma

Euroma is a company that develops and produces a range of spice-based products. The company originated in 1899 as a store selling herbs, spices, and pharmaceutical items (Euroma, 2019).

Euroma reached a top position in the European spice market after acquiring Intertaste in 2018. Production processes of various production sites were united when a new factory in Zwolle opened in 2019 (Euroma, 2022). The factory contains the production and packaging of all dry products, such as herbs and spices, seasonings, and coatings. Euroma also has locations in Nijkerk and Schijndel, which produce and package abient and fresh liquids, respectively (Euroma, 2019). This research focuses on the mixing department at the Zwolle production site.

Introduction to the mixing department in Zwolle

The mixing department in Zwolle holds one of the main processes at this location. This department mixes ingredients to make spice blends. It runs twenty-four-seven to satisfy demand. In the department, ingredients are collected (replenishment), mixed (mixing) and packaged to be sent to the customer or processed further (packaging). Figure 1.1 shows a simplified overview of the mixing process, which we explain in the remainder of this section. We give a more extensive description of the mixing process in Section 2.1.

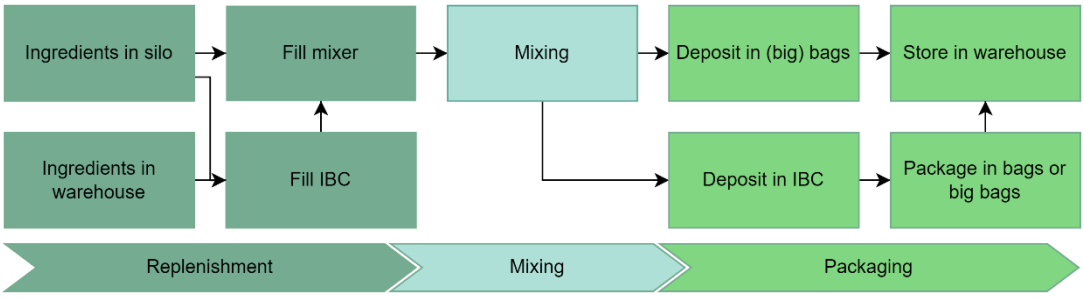


Figure 1.1: Simplified overview of the mixing process, with the steps of replenishment, mixing, and packaging.

Replenishment

The warehouse, internal (inside), or external (outdoor) silos store the ingredients used to fill the mixer. The silos hold the most used ingredients, such as starch and salt. External silos deposit via hoppers into the two largest mixers. Intermediate bulk containers (IBCs) collect the other ingredients. Via hoppers, an internal silo deposits ingredients into an IBC, see Figure 1.2. Ingredients from the warehouse are deposited into IBCs using a filling station.

Figure 1.2: An IBC being filled by a hopper of an internal silo.

Mixing

The mixing department receives the full IBCs. There are eight mixers, with a capacity ranging from 200L to 10,000L. Per mixer, characteristics can differ, such as possible ways to fill them and the kind of product suited for the mixer.

Packaging

When mixed, the product is stored in the same IBCs or directly put in (big) bags, depending on the mixer type. The packaging process packages full IBCs in bags or big bags. Bags weigh around 20kg, and big bags refer to bags of around 400 kg. The warehouse stores the packaged products.

1.2 Problem description

In this section, we briefly describe the problem context in Section 1.2.1 and the causes of the problem and problem statement in Section 1.2.2.

1.2.1 Problem context

Euroma wants to increase the output of the factory. One of the bottlenecks in production is the IBC availability in the mixing department. The IBCs filled with finished products accumulate in the packaging process, stagnating the release of new jobs.

The IBC filling rate for replenishment is 73%, and for packaging, 81%. On average, each mixture (measured over 1,467 jobs) requires 2.0 IBCs.

From filling an IBC until packaging the product in the IBC (if used for packaging), it takes an average of 17.8 hours and 4.3 hours when not using the IBC for packaging (measured over 1,467 jobs).

When used for packaging, the filling rate and the throughput time of an IBC suggest that IBC use can be more efficient. Euroma wants to know how they can increase the availability of IBCs by increasing the filling rate or decreasing the time a job uses an IBC.

Filling rate

In the mixing department, each product follows a standard route. A route consists of the mixer, order quantity, and mixing steps. The process engineer calculates the optimal order quantity by considering limitations such as the minimum order quantity (MOQ) and the incremental order quantity (IOQ), which are contingent on customer agreements. The IOQ

often relies on the size of bags or pallets used to package the finished product, as detailed in Appendix I.

Lead time IBC

The IBC functions as follows: during a specific job, it receives ingredients from the internal silos, bags, and totes. Once it becomes full, it waits until the mixer can receive the ingredients. In the case of specific mixers, the IBC also acts as a temporary storage vessel for the finished product until it is ready for packaging. Cleaning of the IBCs occurs between jobs. Table 1.1 details the average time and quantity of IBCs used for various subprocesses.

Table 1.1: Average throughput times of an IBC per production order (n=806; data week 47 to 49 2022; source: Weekplanning).

Subprocess	Average time (hours)	Average number IBCs used per week
Filling IBC and waiting on mixer	4.25	550
IBC waits for the mixer to finish	1.70	550
IBC waits on packaging	11.37	278
IBC at packaging	0.51	278

The table shows that the IBC spends most of the time waiting on packaging. It is logical to research improvements made in this aspect.

1.2.2 Problem causes and problem statement

The low availability of the IBCs has multiple causes, as identified in the previous sections. Figure 1.3 shows an Ishikawa diagram with the causes (the arrows) and the effect of the problem (the circle) (Graham et al., 1979).

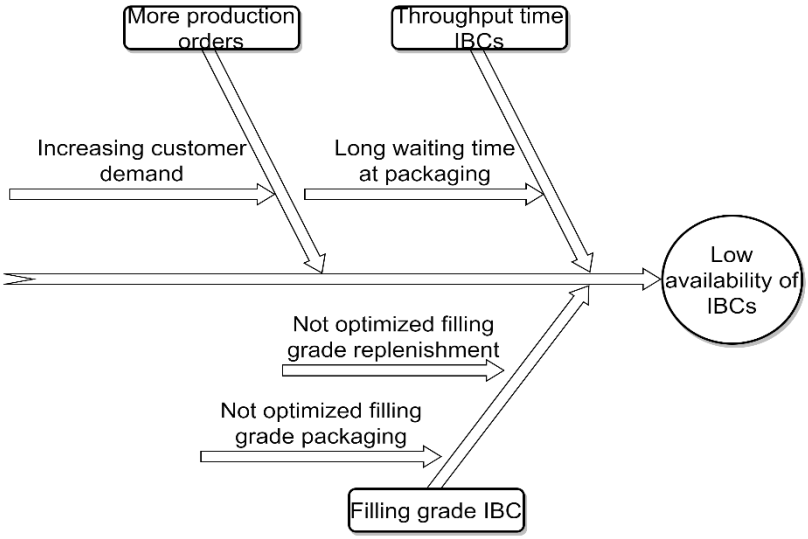


Figure 1.3: Ishikawa diagram with the cause and effect of the problem.

The company works actively towards increasing customer demand, a situation they value greatly. Section 1.2.1 explains that the long lead time of IBCs is primarily caused by the long waiting time at packaging, resulting in being a significant factor in the throughput time of IBCs. Evaluating the filling rate of the IBCs involves analysing the replenishment and packaging processes, both of which remain unoptimized in the current situation. However, the customer agreements regarding the IOQ and MOQ currently limit the possible solutions to make significant changes concerning the filling rate.

With this analysis, we conclude decreasing the long waiting time at the packaging process and increasing the throughput time of the IBCs generate the most impact to improve IBC availability. We define the main objective of the research as finding a solution that contributes to decreasing the lead time at packaging to enhance the availability of the IBCs.

1.3 Research approach

This section gives the scope of the project in Section 1.3.1. Next, Section 1.3.2 and Section 1.3.3 explain the research questions and the methods used to answer them. We set the deliverables of the research in Section 1.3.4. Finally, Section 1.3.5 gives the outline of the report.

1.3.1 Research scope

The focus of this research is the factory of Euroma in Zwolle. The research within this factory is specifically about the mixing department and the use of IBCs. The mixing department contains the processes replenishment, mixing and packaging. We limit the research to the packaging process. Other processes of the mixing department, including cleaning of IBCs, are out of scope.

In addition, we take the following aspects as a given:

- Customer demand
- Mixing schedule
- Ingredients and recipe of a product

1.3.2 Research questions

This section gives the main research question and the sub-research questions, in addition to the research method.

The main research question of this research is:

How can the lead time at the packaging process be decreased to improve the availability of IBCs at Euroma?

We first answer five sub-research questions to find a solution for the main research question.

1. *What is the current situation concerning IBC use at the packaging process of Euroma?*

To answer sub-research question 1, we analyse the context of the core problem. First, we give an overview of the production process. Next, we map the organizational factors concerning the present stakeholders, systems, and data. Finally, we create insight into the performance of the current situation. We collect information by visiting the factory, monitoring the processes, interviewing employees, and deriving information from available data.

2. *What available theory applies to improving the scheduling of the packaging process of the IBCs at Euroma?*

We conduct a literature review to answer the second sub-research question. The literature is relevant to decreasing lead times in comparable processes and general methods used for optimization.

3. *What solutions can potentially contribute to enhancing the availability of IBCs at Euroma?*

We propose multiple potential solutions by combining the knowledge obtained about the current situation regarding IBCs and the existing literature.

4. *Which of the proposed solutions gives the best results for the situation at Euroma?*

This sub-research examines the proposed solutions through simulation testing and evaluation. In this process, we establish performance indicators, create various scenarios to test different solutions and map out the positive and negative implications to ascertain the impact of these solutions.

5. *How can the solution be implemented at Euroma?*

With a formulated solution, we set up an implementation plan to transition ownership of the research to the company.

1.3.3 Deliverables

This research presents the following deliverables to Euroma:

- A solution to contributing to improving the availability of IBCs at Euroma.
- An implementation plan for implementing the proposed solution at Euroma.

1.3.4 Outline of the report

The report follows the following structure. Chapter 2 gives the current situation concerning IBC use at Euroma. Next, Chapter 3 provides the literature review. Chapter 4 describes the modelling and solution approaches. Chapter 5 gives the experiments and their results of the potential solution approaches. Chapter 6 provides advice on how to implement the solution. Lastly, Chapter 7 gives the conclusion and recommendations.

2 Current situation

This chapter answers the sub-research question:

What is the current situation concerning IBC use at the packaging process of Euroma?

We first explain the processes in the mixing department in Section 2.1. Next, we describe the production plan process and the mixing steps in Section 2.2. Section 2.3 and Section 2.4 explain the stakeholders, IT systems, and available data. Finally, Section 2.4 summarizes this chapter.

2.1 Mixing department

The following sections explain the processes within the mixing department. Section 2.1.1 details the replenishment process. Next, Sections 2.1.3 and 2.1.4 provide the mixing and packaging process. Lastly, Section 2.1.5 explains the routing of IBCs.

Figure 2.1 gives an overview of the relationships between steps within and between the different processes of the mixing department. Appendix II shows the enlarged figure.

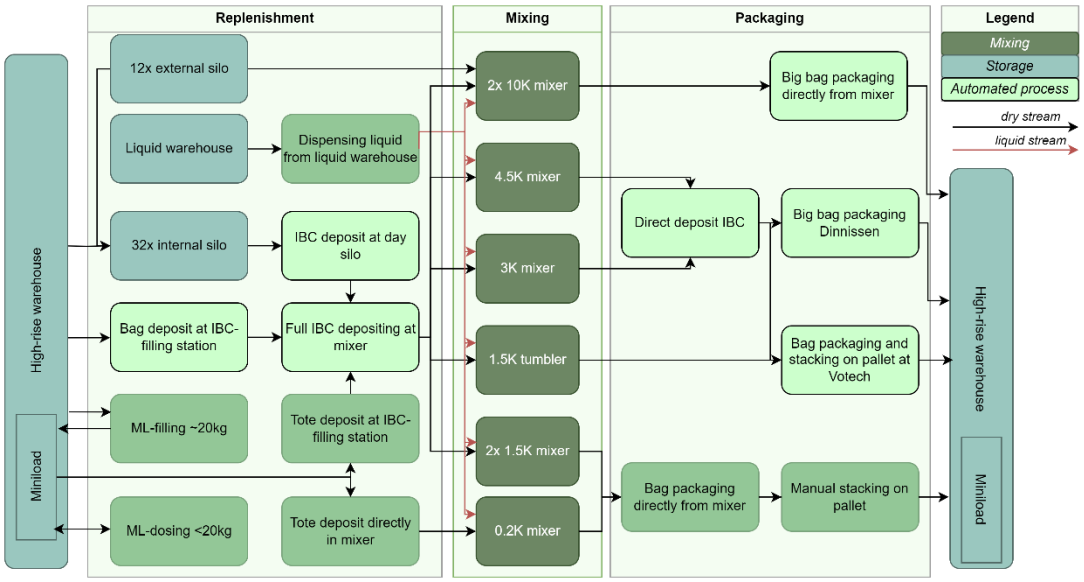


Figure 2.1: Process within the mixing department.

2.1.1 Replenishment of mixer

Outside silos, internal silos, pallets, totes, and barrels if it is liquid store the ingredients needed for the mixtures. See Figure 2.2 for illustration.

Figure 2.2: From left to right: External silos, internal silos, and totes.

For each job, replenishment is necessary. The mixer type and the storage methods of the ingredient determine from which storage to collect from. In this section, we refer to the relevant floor levels in the factory. Figure 2.3 (see 0 for an enlarged version) illustrates the levels.

Ingredients for a mixer consistently come from the largest available carrier. For example, for an ingredient needed in the largest mixer and an ingredient resides in both an internal and external silo, the ingredient always comes from the external silo. The outside silos fill the two largest mixers via a hopper.

IBCs collect ingredients at the internal silos. An Automated Guided (AGV) transports an IBC towards the internal silo depositing point (level 0). If there is still room in the IBC, the IBC travels to the next internal silo or to the IBC filling station (level 0). If it is full, it travels towards a buffer place at the mixer (level 3 for 10K, level 4 for remaining mixers).

At the IBC filling station (level 1), operators fill an IBC using bags or totes. An IBC positions directly under a filling station (level 0), filling one IBC at a time. An AGV collects a full IBC and brings it to the buffer point at the mixer.

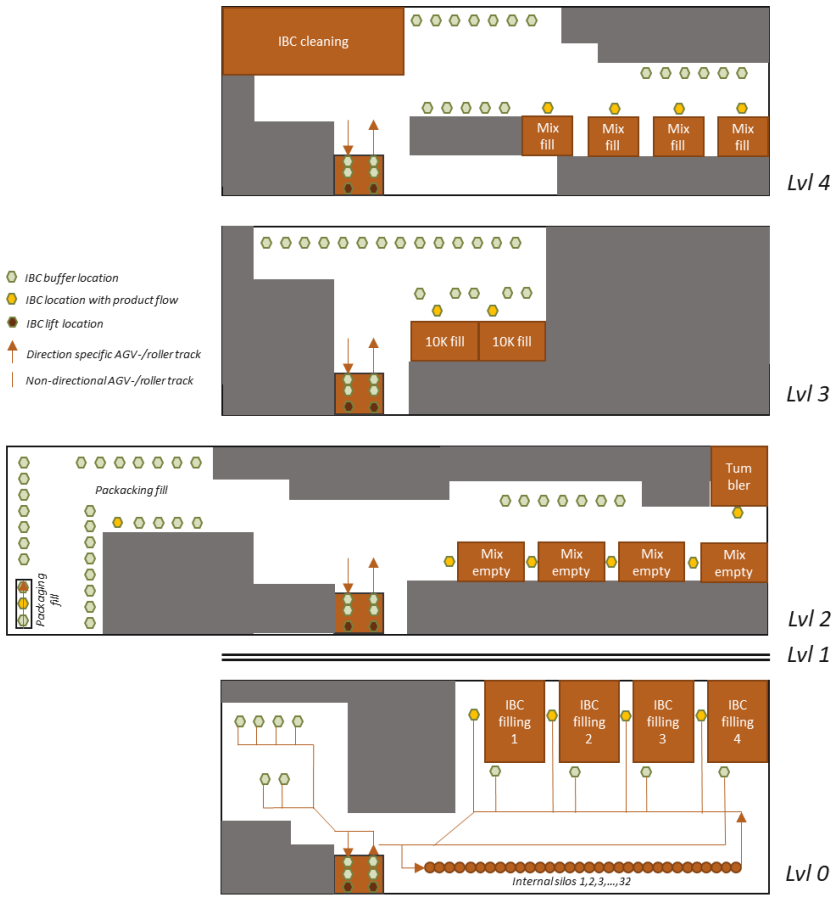


Figure 2.3: Simplified map of the mixing department. Adapted from “Planning & Scheduling for Euroma Zwolle”, by M. Bergman & T. Van Benthem, 03-02-2023.

2.1.2 Mixing

The processing steps of a product indicate when to add which ingredients. Often, there are 1 to 3 processing steps. The first step involves adding most of the dry ingredients, the second includes adding liquid ingredients, and the third covers adding the remaining dry ingredients.

Table 2.1 shows the characteristics of each mixer. The biggest mixer type, the 10K mixer, can hold 10,000L. As a logical consequence, these mixers mix the largest production orders. Outside silos and IBCs fill these mixers. An automated process packages the finished product directly from the mixer into big bags.

Table 2.1: The mixers at Euroma Zwolle and their characteristics.

Type	#	Name	Capacity	Packaging possibilities
10K	2	407 & 408	10,000L	Directly in big bags
4.5K	1	404	4,500L	Via IBC in bag or big bag
3K	2	403 & 405	3,000L	Via IBC in bag or big bag
1.5K	1	402	1,500L	Directly in bags
0.2K	1	401	200L	Directly in bags
1.5K Tumbler	1	409	1,500L	Via IBC in bag or big bag

Only IBCs fill the 4.5K, 3K, and 1.5K mixers. The 4.5K and 3K mixers deposit the product after mixing directly into the same IBCs used for filling. Operators deposit the finished products from the 1.5K mixer into bags. Operators fill the 0.2K mixer by manually depositing totes into it. Packaging takes place directly below the mixer, in the same manner as for the 1.5K mixer.

The 1.5K Tumbler is a unique mixer. One filled IBC contains all the ingredients of the job. The IBC tumbles instead of unloaded into a mixer. Packaging occurs similarly to the 4.5K and 3K mixers. Not all products suit this mixer because it tumbles an IBC and does not use a mixing arm or cutting rotor. For example, powder with liquid does not create a homogeneous product using this mixer. In addition, there is only one mixing step. Hence, all ingredients combine at once.

Figure 2.4: On the left, the Tumbler rotates an IBC. Right, a 10K mixer with pipes above the mixer for ingredient disposal and below for disposal of the finished product.

2.1.3 Packaging

Figure 2.1 and Table 2.1 show that the packaging process depends on the mixer used. The 10K mixers each have their own packaging station with two automatic disposal points for big bags. After mixing, operators deposit the products from the 1.5K and 0.2K into bags and stacked on a pallet.

The IBCs used for replenishment temporarily store the mixed product from the 4.5K and 3K mixers. An AGV transports these, and the IBCs from the 1.5K tumbler, towards a buffer point for one of the two depositing points for the packaging stations. Figure 2.4 displays this area on the left. The machines located below the depositing points. One station is for the Dinnissen, and the other is for the Votech machine. These machines pack the product in respectively big

bags and bags. The packaged bags at the Votech automatically stack on a pallet. Operators operate the machines; this process does not require any other manual labour next to cleaning. Figure 2.5 presents the Dinnissen and Votech machines.

Customer requirements determine the size of the big bag or bag. The warehouse stores the finished product until further processing or shipping to the customer.

Figure 2.5: On the left, the Dinnissen packaging big bags. On the right, the Votech packaging bags

2.1.4 IBC routing

AGVs transport the IBCs. Figure 2.6 shows the flow of IBCs throughout the mixing department. Because of the small size of the 0.2K mixer, this mixer does not use IBCs. The figure represents all other mixers.

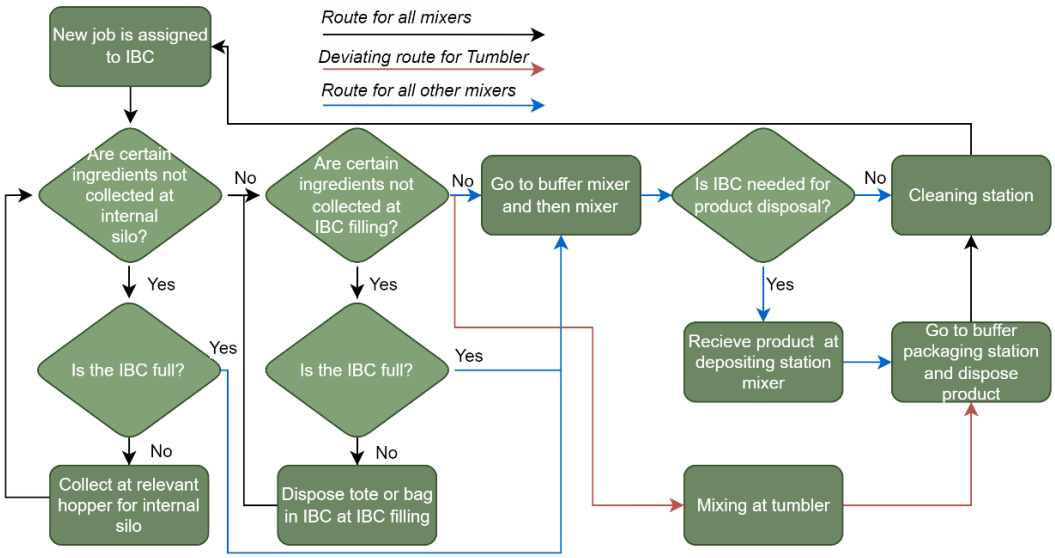


Figure 2.6: Flow of the IBCs through the mixing department

When a new job starts and an IBC is available, these link to each other. During an ongoing replenishment process for a mixer, the first goal is to gather all products from internal silos. As a result, the IBC initially fills up at the internal silo depositing point. If it still has capacity

after completing this task, it collects ingredients from IBC filling stations. Once the IBC reaches its total capacity or collects all the ingredients, it travels to a buffer area next to the relevant mixer.

After mixing, the 4.5K and 3K mixers deposit the finished products into IBCs for packaging. These are the same IBCs used for replenishment since this saves cleaning time. AGVs bring the refilled IBCs to buffer places next to the packaging machines. If all these buffer places are full, the IBC travels to a different level. It is better to steer clear of this scenario to prevent additional handling and strain on shared resources,

Sometimes, replenishment demands more IBCs than packaging, either because the product volume reduces during mixing or due to multiple processing steps with half-full IBCs. When packaging requires fewer IBCs than replenishment, the excess IBCs enter the cleaning station.

The IBCs used for the 1.5K and 10K mixers go to the cleaning station after depositing the ingredients in the mixer. The IBC used for the 1.5K Tumbler mixer goes directly to the packaging station, as disposal is unnecessary.

2.2 Restrictions

The Votech or Dinnissen machines package all products from the 3K, 4.5K, and 1.5K Tumbler mixers. Creating a weekly schedule for the mixers provides knowledge of their anticipated output. However, this is not exact and can be off by hours or days. To address the uncertain mixer output, the order in which the mixed products enter the Votech or Dinnissen machine remains undetermined in advance, with an operator responsible for establishing an optimal sequence.

For both the Votech and Dinnissen, salt rinses the machine after every product change. However, if sequential products differ in specific characteristics, rinsing with salt is insufficient and more extensive cleaning is necessary. To minimize the cleaning time, the operator looks for a series of similar products that require only minimal cleaning. The following characteristics of sequential products determine the degree of cleaning necessary:

- Colour: for example, white to grey requires less cleaning than red to white.
- Allergens: if a product contains one or multiple allergens, such as gluten, mustard, and soja, an extensive cleaning occurs to assure that the next product does not contain these allergens.

The cleaning type for a significant colour change is ‘dry cleaning’ and ‘wet cleaning’ for allergens change. With a dry cleaning, the operator brushes residue away from the machine. With a wet cleaning, the cleaning is more extensive, and the operator uses water and a cleaning product to ensure no allergens are left behind. The dry cleaning takes less time than the wet cleaning. The exact time depends on the machine and ranges from 30 to 120 minutes.

In addition, the sequence is dependent on the halal and kosher claims. The halal and kosher certified products must adhere to Islamic (Eardley, 2014) and Jewish law (Kosher Certifications & Supervisions, 2022), respectively. When a product has one of these certifications, the two products processed before this product must be suitable. A suitable product is a product that does not have the certification but does not contain ingredients forbidden by Islamic or Jewish law.

Finally, specific product characteristics necessitate packaging shortly after mixing. Otherwise, there is a risk of clumping. The time before this happens can differ per product, but it is, on average, 24 hours. This makes the time waiting at the packaging process an important factor.

The understanding of the company is that currently, these rules are not always abided by; in Chapter 5, we confirm this suspicion.

2.3 Stakeholders

This section discusses the most relevant stakeholders to the mixing department and to consider when framing a solution to our research question.

The first stakeholders identified are the Customer Service and Sales departments. Their role in Euroma's agreements with her customers makes them stakeholders. The agreements made with the customers impact the orders' characteristics and profitability.

Next, we identify the operators working with the packaging machines. They work with the machines daily and are a valuable source of information and, therefore, stakeholders. Moreover, the proposed solution in this research may impact their work.

Last, we identify Euroma as a company. The company has a long-term vision to increase sales and production output. This research contributes to realizing a higher potential output for the future.

2.4 IT systems and available data

Multiple systems collaborate to enable the production process. The schedule of the mixers serves as input for the machines in the production area and draws data from multiple systems. Figure 2.7 depicts the relationships between these systems.

Slimstock generates the production forecast and communicates this with LN, the ERP system Euroma uses. In LN, the planners plan the production order for a specific week. An algorithm in Delphi uses the weekly plan in LN and allergen information from PLS PRO to form an optimized schedule for mixing. Delphi communicates this plan with LN. A planner releases the jobs in LN, triggering the job in ESA.

ESA is the most used program in the production area. The interface for the operators and the control in releasing jobs in the replenishment and mixing steps is in ESA. MES is the system that controls the packaging process in the mixing department. ESA and MES interact with DS Automation, the program that controls the AGVs.

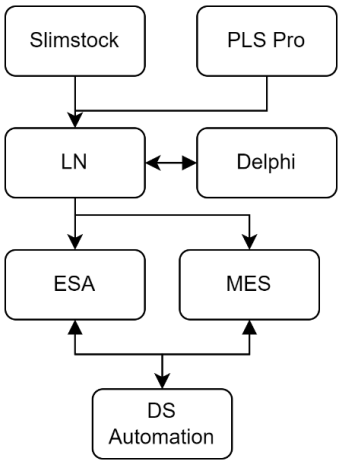


Figure 2.7: Different systems and their relationships.

Multiple systems collect data. Table 2.2 lists the most relevant data to this research.

Table 2.2: Available data and corresponding programs.

Name	Program	Description
Weekly production plan	LN	A weekly plan with the sequence of all jobs planned on each mixer, including the IBCs needed per job
AGV movement	ESA	Data of movements of the AGV and, therefore, the movements of IBCs
Production events	ESA	Data of all transpired events within the mixing department, e.g., times of collecting ingredients, releasing jobs, and mixing
Packaging	MES	Data of packaging at the Votech and Dinnissen
Allergens	PLS Pro	Allergens per product
Product information	LN	Certifications and colour per product

2.5 Summary

This chapter provides answers to the following sub-research question:

What is the current situation concerning IBC use at the packaging process of Euroma?

IBCs collect ingredients to fill the mixers. In addition, IBCs transport finished products to packaging. The fill rate of the IBCs depends on the quantity made and the division of ingredients over the IBCs. The planners determine the quantity made, restricted by the MOQ and IOQ.

When using the IBCs for packaging finished products, it is necessary to consider the following restrictions for the processed product sequence:

- Cleaning (dry) is necessary between products if the colour changes.
- Extensive (wet) cleaning is necessary after a product without allergens follows a product with these allergens.
- A kosher certified product can only be processed if the two previous products are either kosher certified or kosher suitable.
- A halal certified product can only be processed if the two previous products are either halal certified or halal suitable.
- The finished product cannot stay in an IBC for an extended period. The time differs per product but is, on average, 24 hours.

3 Literature review

In this chapter, we provide an overview of the literature related to the problem described in the first two chapters. First, Section 3.1 covers the theory related to planning and scheduling, with as goal reducing the lead time of IBCs in the packaging process. In Section 3.2, we describe multiple solution approaches. This section first provides methods for reducing the lead time, such as constructive and improvement heuristics. In addition, it gives multiple operators and neighbourhood structures used in the solution approaches. Finally, Section 3.3 provides a conclusion of the sub-research question:

What available theory applies to improving the scheduling of the packaging process of the IBCs at Euroma?

3.1 Planning and scheduling

In this section, we zoom in on the packaging process, where there is always a queue of IBCs for the Votech and Dinnissen machines. The goal is to find a method to reduce this queue and the lead time of IBCs.

First, we look at the broader production planning process in Section 3.1.1. Section 3.1.2 gives information about the machine scheduling problem. Section 3.1.3 explains the difference between online, offline, and real-time scheduling. Lastly, Section 3.1.4 provides methods to incorporate sequencing restrictions.

3.1.1 Production planning

The approach of hierarchical production planning incorporates a series of models on different hierarchical levels throughout an organization. Figure 3.1 shows a hierarchical planning scheme. The highest organizational level decides what item is produced in which factory. Next, aggregated planning determines production and inventory levels per plant. The sales forecast in the mid-term leads to a production plan. Next, the approach suggests scheduling product families (items with similar characteristics) before individual items. At the lowest level, the scheduling of parts and components occurs, with each lower level constrained by decisions made at higher levels (Vollmann et al., 1992).

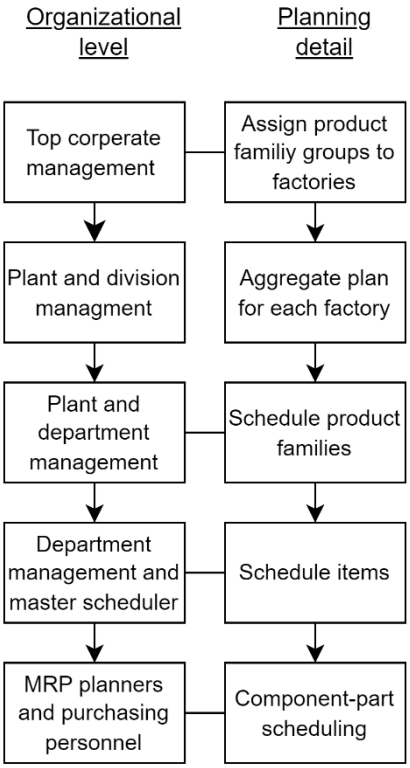


Figure 3.1: Hierarchical production planning scheme (Vollmann et al., 1992).

3.1.2 Machine scheduling

Scheduling involves assigning resources to tasks within specified timeframes, with the aim of optimizing one or more objectives. In machine scheduling, the resources are machines, denoted by i. j denotes the tasks, or jobs, on the machines. Each job relates to information about its processing time (p_{ij}), release date (r_j), due date (d_j), and weight (w_j). The latter is a priority factor, denoting the importance of the job (Pinedo, 2016).

Assumptions often used in scheduling models are the following (Haupt, 1989):

- One machine only processes one job at a time.
- Each machine only processes one job at a time.

- Each job must finish without interruption.
- Each machine is continuously available.
- The limiting resource is the machine, not other shared resources.

A frequently used notation for machine scheduling is the three-field notation: $\alpha|\beta|\gamma$. This notation holds the machine environment (α) processing characteristics and constraints (β), and the objective function (γ) (Graham et al., 1979).

Machine environment

Machine scheduling problems typically classify as single-stage or multi-stage scheduling problems. A single-stage machine scheduling problem involves only one operation for each job. Multi-stage scheduling problems involve jobs requiring operations on different machines (Anderson et al., 2003).

This research only looks at the packaging step, with only one activity. Hence, we classify the problem as a single-stage machine scheduling problem. Within this category, Pinedo (2016) distinguishes the following categories:

- Single machine: 1 machine and n jobs.
- Identical machines in parallel: m identical machines and n jobs.
- Machines in parallel with different speeds: m machines and n jobs, where the processing time of a machine differs with a constant factor.
- Unrelated machines in parallel: m machines and n jobs, where the time to complete a job is different per combination of job and machine.

Processing characteristics and constraints

Pinedo (2016) distinguishes multiple processing characteristics and constraints. A problem may involve more than one restriction or constraint. A list of processing characteristics and constraints suitable for a single-stage machine schedule problem is the following:

- Release dates: the job cannot start before the release date r_j.
- Preemptions: the job can be interrupted to appoint another job to the machine.
- Precedence constraints: completion of one or more jobs before another job starts processing.
- Sequence dependent setup times: in between jobs j and k, there is s_{jk} set-up time.
- Job families: a machine can process jobs from the same family sequentially on a machine without requiring setup between jobs.
- Batch processing: a machine can process several jobs simultaneously.
- Break downs: machines are not continuously available, also known as machine availability constraints.
- Machine eligibility restrictions: not all machines can process all jobs. The set M_j denotes the set of machines that can process job j.

Objective functions

The main groups of objective functions distinguished by De Souza et al. (2022) are:

- classic due date related criteria,
- makespan and completion time related criteria,
- and multi-criteria and additional objective functions.

Classic due date related criteria depend on the due date d_j. C_j denotes the completion time of job j, which is the latest time the job is in the system. This determines the lateness of a job, L_j:

$$L_j = C_j - d_j \quad (1)$$

Alternatively, the tardiness of a job, T_j, indicates lateness, which cannot be negative.

$$T_j = \max(C_j - d_j, 0) \quad (2)$$

Lastly, a unit penalty, U_j associated with a tardy job (Pinedo, 2016):

$$U_j = \begin{cases} 1 \; if \; C_j > d_j \\ 0 \; otherwise \end{cases} \quad (3)$$

The makespan, C_{max}, is the time that it takes to complete all jobs. The processing times of the jobs, and the times the machine is not working (e.g., waiting for a job and set-up time) determine C_{max} (Lee et al., 2010). Other time related criteria include weighted shortest processing time and minimization of the set-up time (De Souza et al., 2022).

Multi-criteria optimization problems involve optimizing more than one criterion or objective. If these criteria conflict with each other, a trade-off is necessary (Červeňanská, et al., 2021). Barnes & Vanston (1981) provide an example where they minimize the sum of delay penalties and setup costs. Methods for dealing with multi-criteria optimization include (Lalla, 2022):

- Weighted scoring method: The individual objectives can receive a fixed or dynamic weight for prioritizing specific objectives above others (Červeňanská, et al., 2021).
- ϵ-constraint method: One objective is optimized, and the remaining objectives change into constraints (Miettinen, 2004).
- Lexicographic method: The objectives are sorted on importance, after which the optimization problems are solved one by one, where the most important objective is solved first. After each problem is solved, we add the restriction that the previously optimized objectives cannot be worse than the value found in its optimization (Arora, 2017).
- Goal programming: Goal programming assigns specific goals to individual objectives and establishes a mathematical model aimed at minimizing deviations from these set goals. (Arora, 2017).

Additional objective functions can include the minimization of total earliness and total late work (De Souza et al., 2022).

3.1.3 Online, offline, and real-time scheduling

In offline scheduling, advance knowledge of all scheduling data, including processing times, release dates, and due dates, allows for creating the entire job schedule at time zero. In online scheduling, this data is unknown beforehand. A specific online scheduling problem is the online-over-time scheduling problem, where jobs release at different times, and the future quantity of jobs to be released and their respective release dates remain unknown (Pinedo, 2016).

Real-time scheduling involves creating a schedule during run-time that meets strict timing constraints. It differs from online scheduling in that it often involves periodic data releases, where a recurring task uses input from this data to create a new schedule. This process of dynamically adjusting the schedule based on incoming data is crucial in real-time scheduling, where timely response to events is essential. For example, in real-time systems such as air

traffic control or autonomous vehicles, delays in scheduling decisions could have serious consequences (Goossens & Richard, 2004).

3.1.4 Sequencing restrictions

Recall that Section 2.3, explains that a halal certified product precedes at least two halal suitable or certified products, and the same applies to kosher products. A proposed solution needs to account for these restrictions to be considered feasible.

Sun et al. (2010) name three methods for reaching a feasible option when there are sequencing constraints.

- Reject the proposed solution when it is infeasible.
- Repair infeasible solutions to make them feasible.
- Improve operators to guarantee a feasible solution.

Modification of the solution does come with its disadvantages. These include being computationally intensive and limiting the solution space. Therefore, Sun et al. (1998) propose an alternative method by penalizing infeasible solutions, making selecting a feasible solution more probable.

3.2 Solution approaches

Combinatorial optimization deals with optimising an objective function over a finite domain. Often, integers and natural numbers restrict variables. These and other restrictions result in a limited number of feasible solutions. The single-machine scheduling problem is a combinatorial optimization problem (Ramalhinho-Lourenço, 2019). Exact or approximation methods find utility in solving combinatorial optimization problems. (De Giovanni, 2017).

Exact methods aim to find the optimal feasible solution. These methods include branch-and-bound algorithms, dynamic programming, and integer (linear) algorithms (Reddy, 2019). However, there are instances when we cannot employ exact methods. This may arise due to the problem's complexity, where an exact method fails to yield an optimal feasible solution, or because of time constraints that hinder the search for a solution. Approximation methods use heuristics to find a good, not necessarily optimal, feasible solution in a reasonable amount of time (De Giovanni, 2017).

3.2.1 Algorithms

Table 3.1 provides multiple algorithms used to solve combinatorial optimization problems. The choice of algorithm depends on the size and complexity of the problem, as well as the desired level of optimality of the solution (Tahami & Fakhraver, 2022).

Table 3.1: Algorithms for combinatorial optimization problems (Tahami & Fakhraver, 2022).

Category	Example of algorithms
Fitting algorithms	Next fit, first fit, and best fit
Exact enumerative methods	Branch-and-bound and branch-and-price algorithms
Basic approximation algorithms	Near-optimal algorithms and efficient approximation schemes
Heuristics	Local search algorithms, greedy procedures, simulated annealing, and GRASP algorithms
Hybrid approaches	Metaheuristics and combining Lagrangian relaxation and column generation

The theory does not recognize one method as best for solving combinatorial optimization scheduling problems. In Section 4.1, we conclude that the problem is a real-time scheduling problem. In most existing methods for time-restricted scheduling, such as real-time or online

scheduling, the theory employs heuristics, primarily due to the intensive computational time required by conventional search and optimization methods, making them less favoured for time-restricted scheduling (Ang et al., 2009). The remainder of this section gives multiple constructive and improvement heuristics to solve the scheduling problem at Euroma.

Constructive heuristic

A constructive heuristic starts with an empty solution and iteratively adds one element to the solution by applying some specific selection criterion (De Giovanni, 2017). In the real-time scheduling problem at Euroma, we have the practical requirement for the algorithm to make fast decisions. Therefore, the constructive heuristics mentioned next are relatively simple algorithms.

First-come-first-serve

The first-come-first-serve (FCFS) heuristic schedules the jobs in order of arrival, where the first arrived processes first. This method is straightforward and reduces the variation in waiting time between jobs. However, it does not consider other performance criteria, such as minimizing set-up time, resulting in a higher average waiting time (Ahmad, 2023).

Nearest neighbour heuristic

In specific cases, solving a single-machine scheduling problem with sequence-dependent set-up times aligns with the approach of solving a travelling salesman problem (TSP) (Minetti, et al., 2001; Mustu & Eran, 2018). In the TSP, the assignment is finding an optimal route between a given set of cities. A relatively simple heuristic used for TSP is the nearest neighbour heuristic (NNH). Here, the city closest to the last added city becomes the next addition. In machine scheduling, the cities represent jobs, and the nearest neighbour corresponds to the job that contributes most positively to the objective and repeats for all jobs. Because of its simplicity, this is a fast algorithm. However, the solutions are not necessarily close to the optimum (Oliveira & Carravilla, 2009). Moreover, additional constraints or goal functions can disqualify this heuristic from applying to the single-machine scheduling problem with sequence-dependent set-up times.

NEH algorithm

A well-known constructive heuristic in the scheduling problem is the Nawaz, Enscore, Ham (NEH) algorithm (Nagano & Miyata, 2016). This algorithm first sorts the jobs in descending order of processing time. The first job on the list, the one with the largest processing time, is scheduled in an empty schedule. Next, the second job on the list is added by trying to insert it in every possible place in the existing sequence and calculating the objective function value. The schedule with the best objective function value is selected. This repeats until the scheduling of all jobs (Nawaz et al., 1983). More complicated metaheuristics do not outperform this algorithm (Ruiz & Stützle, 2007). Moreover, this algorithm proves effective scheduling for the mixers at Euroma (Benthem, 2021). However, it is more computationally expensive than the aforementioned algorithms (Nagano & Miyata, 2016).

Improvement heuristic

The goal of the improvement heuristic is to find the best solution among many possible solutions. Often, a local search finds neighbouring solutions that give a better objective than the current solution. The set of all solutions obtained by making a small change to the current solution defines the neighbouring solutions (Cormen et al., 2009), see Section 3.3.3 for more information. One of the simplest methods for minimizing a function is the steepest descent (Rosing et al., 1998). Moreover, simulated annealing (SA), tabu search (TS), and genetic

algorithms (GA) are popular improvement heuristics applied to machine scheduling problems (Ruiz & Vázquez-Rodríguez, 2010; Cinar et al., 2015).

Steepest descent

Steepest descent searches for the best neighbour in the complete neighbourhood of the initial solution. This best neighbour is accepted if it does not decrease the objective function value. Iterations continue until reaching a local optimum or a stopping criterion. It is a very simple and quick algorithm. However, it has no method of getting out of a local optimum, often preventing it from finding the optimal solution (Winston, 2004).

Tabu search

Tabu search (TS) uses a local search method where local optima are escaped by remembering already explored solutions in a tabu list. In addition, a set of aspiration criteria allows the algorithm to make exceptions to the tabu list when finding a better solution. The search stops when meeting a stopping criterion, such as reaching a maximum number of iterations. The best solution found during the iterations is returned as the result (Glover, 1989).

A long tabu list can restrict the algorithm's ability to explore promising areas of the search space, reducing the diversity of solutions generated. Therefore, the algorithm may get stuck in suboptimal solutions and fail to find better alternatives. Moreover, by keeping solutions marked as tabu for an extended period, the algorithm may struggle to adapt to changes made in the solution. When previous moves become more favourable, the tabu list prevents the algorithm from revisiting them, preventing the discovery of improved solutions.

A small tabu list can cause premature convergence by quickly exhausting available moves and revisiting solutions frequently, resulting in the algorithm settling for suboptimal solutions too early. In addition, the limited diversification capacity of a small tabu list narrows the exploration of the solution space, reducing the chances of finding globally better solutions and keeping the algorithm stuck in suboptimal regions. Moreover, it may lead to cycle formation, where the algorithm repetitively revisits the exact solutions without making significant progress, hindering the search for better solutions. Lastly, a small tabu list may not provide enough opportunities to escape local optima, causing the algorithm to get trapped in suboptimal points and failing to reach better global solutions (Salhi, 2002).

Simulated annealing

Simulated annealing (SA) starts with an initial solution and iteratively explores the neighbourhood by making random changes to the current solution. Based on a temperature parameter, the algorithm accepts moves that lead to better solutions and moves that lead to worse solutions with a certain probability. When the temperature is high, relatively more worse solutions are accepted. As the algorithm progresses, the temperature gradually reduces, allowing the algorithm to focus more on exploiting the best solutions found so far. As with TS, SA iterates until it meets a stopping criterion and the best solution is returned (Kirkpatrick et al., 1983).

Genetic algorithms

Genetic algorithms are optimization algorithms inspired by natural selection and genetics. The algorithm creates a population of potential solutions and iteratively evolves the population over many generations. In each generation, the algorithm evaluates each individual's fitness based on a given objective function and selects the best individuals to reproduce and create the next generation. The algorithm creates new individuals that combine the traits of the

selected individuals. The selection, reproduction, and variation processes repeated for many generations, allowing the algorithm to change into optimal solutions (Holland, 1992).

3.2.2 Operators and neighbourhood structures

Local search uses neighbourhoods of the current solution to find improved solutions. All solutions to a problem lie within the solution space S, neighbourhood S_i is part of S that only includes the neighbourhood of solution i. The set of neighbourhood operators defines the size of the neighbourhood. The neighbourhood should be connected. This means any initial solution can iterate to any other solution in S (Radar, 2010).

Move, swap, K-opt, and OR-opt are operators frequently used in optimization algorithms for scheduling problems or TSP (Radar, 2010).

- Move: The move operator removes a job from the current schedule and relocates it to a new position, causing the other jobs to shift up or down to accommodate the move.
- Swap: The swap operator involves swapping the positions of two jobs within the schedule without changing the positions of any other jobs.
- K-opt: K-opt refers to a move operator that involves removing k connections from the current schedule and reconnecting the separate parts to form a new schedule.
- OR-opt: The OR-opt algorithm analyses every possible subsequence of s jobs within the given schedule, checking whether inserting them between two other jobs results in a lower cost. When finding a solution with a lower cost, the operator replaces the original sub sequences. Otherwise, it reduces the size of s by one.

One or multiple jobs, positions, or connections need to be selected to use an operator. This should be selected with care, as this can determine if a neighbourhood is connected or if a schedule is feasible (Zhang et al., 2018). For move, we need to select a job and position. The swap operator requires selection of two jobs. The selection of the jobs depends on the heuristic. Regarding K-opt and the number of connections to choose (k), a smaller k leads to fast iterations. With larger k the algorithm becomes slower, but the solutions are generally of a higher quality (Lin & Kernighan, 1973).

3.3 Conclusion

The literature review's main objective is to address the second sub-research question:

What available theory applies to improving the scheduling of the packaging process of the IBCs at Euroma?

To answer this question, we provide literature related to improving the filling rate of the IBCs and solution approaches. This section answers the sub-research question by combining information from this and the previous chapters.

Section 3.1 gives literature on reducing the lead time of IBCs. First, it explores the overall production planning process. Relating this to findings of Section 2.1 we find that the scheduling at packaging can be placed at the lowest level in the hierarchical production planning scheme. Where the output from the mixers constrains the decisions possible at this level.

Section 3.1 continues by discussing the different types of problems related to machine scheduling. The packaging operations involve using bags and big bags, which renders the sets of jobs unexchangeable between the machines. Consequently, we classify the problem as two

separate single-machine scheduling problems. Our findings in Section 2.2 indicate that the packaging process can only begin after completing the preceding steps, and the finished products can remain in an IBC for a limited period. Therefore, the packaging problem involves release and due dates. The change in colour and allergens between two sequential jobs influences the cleaning time, related sequence-dependent set-up time.

We differentiated between online, offline, and real-time scheduling. Given the dependence of the packaging process on the progress of preceding processes (mixing) and transportation, we cannot precisely determine when the IBCs will arrive at the packaging step. However, we do have an idea of the events about to happen in the preceding step. Therefore, an offline scheduling technique that responds to real-time information is more appropriate for our research than a completely online scheduling technique.

Section 3.1 provides multiple ways to deal with restrictions; these include rejecting infeasible solutions, repairing infeasible solutions, improving operators, or penalizing infeasible solutions.

Section 3.2 gives solution approaches for the described problems. In Section 3.3.1, we find solution approaches for reducing the lead time of IBCs, where the main focus is on constructive and improvement heuristics that give solutions in a reasonable amount of time. We explain different operators and neighbourhood structures. The operators and relevant parameters must be carefully selected to realise a connected neighbourhood.

4 Modelling and solution approaches

In this chapter, we explain the modelling approach and multiple approaches for decreasing the stagnation of IBCs in the packaging process. Section 4.1 outlines the modelling approach. Next, Section 4.2 gives the modelling assumptions and simplifications. Section 4.3 outlines the model description. Section 4.4 explains the parameter calculations, which connect the practical situation to the model. Section 4.5 defines the objective function. Section 4.6 shows multiple solution approaches. Lastly, Section 4.7 provides the summary of this chapter and answers the following sub-research question:

What solutions can potentially contribute to enhancing the availability of IBCs at Euroma?

4.1 Modelling approach

Per machine, there is one queue. The machines work constantly, without exactly knowing when a job arrives; therefore, solving the problem in real-time is appropriate. The queues are primarily independent of each other. However, they are connected by sharing the same limited number of buffer places before the packaging machines. To treat the problem as separate single-machine scheduling problems we use an input variable representing the state of the buffer (see Section 4.3).

The processing times of the jobs are job dependent. The change in colour and allergens between two sequential jobs determine the cleaning time. In addition, the second last processed job is relevant, as the scheduling of certified jobs depends on the two last processed jobs. These factors translate to a set-up time dependent on three sequential jobs.

4.2 Modelling assumptions and simplifications

The following statements outline the assumptions and simplifications we rely on to establish the model. These limit the complexity of the problem for the sake of limited available time.

1. The processing time of a job is static and not dependent on, for example, available AGVs or operators. These shared resources are always available. The processing time includes the travel time.
2. There are no machine breakdowns, and there is no maintenance needed.
3. We assume that all products can stay in an IBC for a maximum of 24 hours before any negative impact on the product is involved.
4. The model only uses information about jobs in the mixers or further in the process.

4.3 Model description

The model is a single-machine scheduling problem with release dates, due dates, and sequence-dependent set-up times.

Following the notation from Section 3.1.2, there is a set of J jobs to be scheduled on one machine. Every job $j = -1,0,1 \dots, J$, has a release date r_j, processing time p_j, and due date d_j. Jobs -1 and 0, are the last two processed jobs. The properties of the last two processed jobs influence the set-up time, resulting in their inclusion in the schedule. Their positions on the schedule remain fixed since they are either already processed or undergoing processing. The set-up time for job j, which follows job k, which follows after job l, is $s_{l,k,j}$. IBC_j detnotes the number of IBCs of job j at the packaging step.

S_i gives the order the jobs are placed in, where $i = 1., \dots, I$ represents different schedules. $S_{i,q}$ refers to the q'th job on the schedule, and the last job as $S_{i,Q}$: $S_i = \{S_{i,-1}, S_{i,0}, \dots, S_{i,q}, \dots, S_{i,Q-1}, S_{i,Q}\}$. The model schedules all jobs; therefore, the number of jobs in the

queue equals the number of jobs in the scheduled, $Q = J$. Dummy jobs are excluded from this definition but are instead incorporated in the set-up time; see Section 4.4.1.

A job cannot start before its release date and the finishing time of the previous job. In addition, during the set-up time $s_{l,k,j}$, which follows directly after job k has finished, no jobs can be processed. The start time of job j is h_j and is dependent on the decision variable S_i.

We define the objective as a combinatorial optimization problem, with as goal to minimize the makespan and tardiness, curtail a full buffer, and prevent the use of dummy jobs. Section 4.5 describes the objective function in more detail. The objective uses the binary variable X, which indicates if the buffer before packaging is full, meaning storing IBCs on different floors. $X = 1$ if the buffer is full and $X = 0$ if not.

4.4 Calculation parameters

The model uses parameters calculated by using the information of the process. This is dependent on the situation at the company.

4.4.1 Set-up time

The setup time, $s_{l,k,j}$, depends on the difference in allergens, colours, and certifications of the sequential jobs l, k and j.

In the context of scheduling with certifications, there are instances where the available jobs cannot create a feasible schedule because suitable jobs for transitioning between non-suitable and certified jobs are lacking. The parameter $e_{j,cer}$ gives information on all the certifications $cer \in CER$ of job j, where $e_{j,cer}$ is 'Certified', 'Suitable', or 'Not suitable'. Machines can process dummy jobs to create a feasible sequence if no suitable jobs are available before producing a certified job. There are two possible dummy jobs: 'salt' and 'stop'. In a situation where only one suitable job is available, a dummy job called 'salt', which processes salt similarly to a regular job, can be added to the schedule. This purifies the machine in the same way as it would have been with a regular job. However, having two salt jobs in a row is not allowed. Therefore, if no suitable job is present and no non-certified jobs are left to be processed, the only solution is to halt production and wait for suitable jobs to become available. In this situation, a dummy job named 'stop' is added to the schedule.

Given these requirements for the dummy jobs, we incorporate three jobs in the sequence dependent set-up time, where the setup time is dependent on the sequence of job j, after k, after l. The time needed for the dummy job 'salt' is 45 minutes. The time for 'stop' is set to 24 hours. This leads to a very high set-up time, preventing it from being scheduled when there are other options, aligning with the penalty strategy from Sun et al. (1998).

$$s_{l,k,j}^{cer} = \begin{cases} 24 \text{ hours} & \text{if } e_{k,cer} = \text{'Not suitable' and } e_{j,cer} = \text{'Certified'} \\ 0.75 \text{ hours} & \text{if } e_{l,cer} = \text{'Not suitable', } e_{k,cer} = \text{'Suitable', and } e_{j,cer} = \text{'Certified'} \\ 0 & \text{otherwise} \end{cases}$$

The set-up time also needs to incorporate the cleaning time. Binary value $dry_{col1,col2}$ sets to 1 if between $col_1 \in COL$ and $col_2 \in COL$ a dry cleaning is necessary, where COL contains all the colours products can have. c_j holds the colour of job j. Binary value $b_{j,a}$ is sets to 1 if job j contains allergen a and 0 otherwise, where $a = 1, \dots, A$, and A the total number of allergens. t_{dry} and t_{wet} represent the cleaning times.

$$s_{k,j}^{clean} = \max\left(dry_{c_k,c_j} * t_{dry}, \max_{a \in A}\left(\max(0, b_{k,a} - b_{j,a})\right) * t_{wet}\right)$$

In addition, the machine often holds residues of the previous product. This can affect the flavour or colouring of the following product. Therefore, the operator rinses with salt to cleanse the machine (less than needed for a 'salt' order). When the jobs contain the same product, this is not necessary. Moreover, when the colour changes, additional brushing removes more pigment. These small cleanings are unnecessary when already conducting dry or wet cleaning. m_j denotes the product of job j.

$$s_{k,j}^{product} = \begin{cases} 15 \text{ minutes} & \text{if } m_j \neq m_k \text{ and } s_{k,j}^{clean} = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$s_{k,j}^{colour} = \begin{cases} 10 \text{ minutes} & \text{if } c_j \neq c_k \text{ and } s_{k,j}^{clean} = 0 \\ 0 & \text{otherwise} \end{cases}$$

These four values together add up to make the sequence-dependent set-up time.

$$s_{l,k,j} = \max_{\text{cer} \in \text{CER}} \left(s_{l,k,j}^{cer}\right) + \max\left(s_{k,j}^{clean}, s_{k,j}^{product} + s_{k,j}^{colour}\right)$$

4.4.2 Release date, due date, start date, and status

Table 4.1 gives an overview of the values given to parameters r_j, d_j, h_j, and $status_j$.

The status of job j, $status_j$, is 'Planned' when the job is still in the mixer. r_j is equal to two hours after starting the mixer. When the IBCs are at the buffer, the status changes to 'Released' and r_j updates to this time. When the job status is 'Active', the place in the schedule is final. When job j arrives at the machine, h_j changes from the planned starting time to the realized starting time. After packaging, the status changes to 'Stopped'.

The start date is restricted by the release date of the job and the finishing and set-up time of the previous job.

$$h_{S_{i,q}} = \max\left(r_j, h_{S_{i,q-1}} + p_{S_{i,q-1}} + s_{S_{i,q-2},S_{i,q-1},S_{i,q}}\right)$$

Table 4.1: Job flow and parameter definition.

Job place	$Status_j$	Release date r_j	Due date d_j	Start date h_j
In mixer	Planned	Estimated: 2 hours after start mixing	Estimated: 26 hours after start mixing	Estimated: according to the schedule
At buffer	Released	Arrival time at buffer	$r_j + 24$ hours	Estimated: $h_{S_{i,q}}$
Packaging	Active	Arrival time at buffer	$r_j + 24$ hours	Start date packaging
Finished	Stopped	Arrival time at buffer	$r_j + 24$ hours	Start date packaging

4.4.3 Full capacity of the buffer

If the buffer reaches capacity, the binary value X is 1. Calculating this, involves jobs of all machines, as the buffer is shared. Recall from Section 2.1.4 that when the buffer is full, IBCs are placed on other floors, giving additional strain to the resources. g is of the set G, which contains all machines. $IBCpresent_g$ represents the number of IBCs physically present at the queue for machine g, excluding the jobs which are still in the mixer or already left the queue. To make this distinction, we use the variables $status_j$ and IBC_j, which holds the number of IBCs used by job j. The variable $IBCfull$ captures the total number of IBC buffer places available at the packaging machines.

$$IBCpresent_g = \begin{cases} IBC_j & \text{if } status_j = \text{'Released'} \\ 0 & \text{otherwise} \end{cases} \qquad \forall\, g \in G$$

$$X = \begin{cases} 1 & \text{if } \sum_{g=1}^{G} IBCpresent_g \geq IBCfull \\ 0 & \text{otherwise} \end{cases}$$

4.4.4 The schedule

The model's output is the schedule with the lowest objective function value. To use this schedule in practice, it is necessary to incorporate the dummy jobs and cleaning tasks into the schedule. The cleaning tasks take place before the processing of dummy jobs. Table 4.2 illustrates an example of a final schedule.

Section 4.4.1 gives the calculation for the start time for the jobs. However, it excluded the calculation for the starting date of cleaning and dummy jobs.

The start date for the cleaning job between job k and j is calculated by $h_k + p_k$. This is a wet cleaning when $\max_{a \in A}(\max(0, b_{k,a} - b_{j,a})) > 0$ and a dry cleaning otherwise. In the example, a wet cleaning takes 2 hours and a dry cleaning 1 hour. The starting time of a dummy job is $h_j + p_j + s_{k,j}^{clean}$.

Table 4.2: Example of a final schedule, including dummy and cleaning jobs.

Job	Product	Start date	Allergen	Colour	Halal	Kosher
J1	64600	24-03-2023 16:26		Red	Suitable	Suitable
J2	60931	24-03-2023 19:00	A1	Red	Not suitable	Suitable
CleanDry		24-03-2023 19:44				
J3	62038	24-03-2023 20:44	A1, A2	Brown	Not suitable	Suitable
J4	62040	24-03-2023 21:08	A1, A2	Brown	Suitable	Suitable
CleanWet		24-03-2023 21:32				
DummySalt		24-03-2023 23:32				
J5	55422	24-03-2023 00:17	A2	Orange	Certified	Not suitable
J6	59884	25-03-2023 00:59	A2, A3	Brown	Certified	Suitable
J7	59884	25-03-2023 01:45	A2, A3	Brown	Certified	Suitable

4.5 Objective function

In this problem, there are multiple (competing) objectives. This section first gives the different objectives. Next, these objectives are combined.

Objectives

The main goal is to minimize the time of IBCs in the packaging process. This aligns with the objective of minimizing the makespan of jobs, as stated in Section 3.2. The makespan accounts for processing, machine cleaning, and machine idle time for jobs finishing in the mixers. The time the last job, $S_{i,Q}$, is planned to start, plus its processing time forms the makespan.

$$F_{Makespan} = h_{S_{i,Q}} + p_{S_{i,Q}}$$

Using dummy jobs is costly in terms of production time. Therefore, we want to delay using dummy jobs until there is no other possibility. Hence, we would rather not place them early in the schedule. An incorporated penalty when the first jobs in the schedule use dummy jobs causes postponement of the use of dummy jobs. Penalizing only the first jobs that use dummy jobs offers several advantages. Firstly, it improves production time efficiency by avoiding excessive penalties on all occurrences of dummy jobs. As using dummy jobs is time-consuming, limiting the penalty to only a few instances helps maintain overall efficiency while ensuring feasible schedules. Secondly, this approach preserves flexibility in optimization. By penalizing only r initial jobs that use dummy jobs, the scheduling algorithm can make decisions for the near future without overly constraining other optimizations later in the

schedule. This balance between feasibility and flexibility allows for adaptability to future jobs, resulting in better overall scheduling outcomes.

Multiplying the duration of the dummy job by a factor that decreases as the job's position in the schedule is further in the future defines the penalty. We set this factor to $r + 1$ minus the place of the job; the first processed job has a factor of $r + 1 - 1$, and the r'th on the schedule $r + 1 - r = 1$. This approach leads dummy jobs to be used further in the future, rather than close, where more suitable jobs are available.

$$F_{Dummy} = \sum_{q=1}^{r} s^{cer}_{S_{i,q-2},S_{i,q-1},S_{i,q}} * (r + 1 - q)$$

Minimizing the tardiness of individual jobs is necessary, as products cannot remain too long in an IBC. Tardiness happens when jobs exceed a specified time limit, as Section 3.1.2 outlines. After the due date passes, the job needs to be processed as fast as possible. A commonly used method to deal with this is squared tardiness (Sun et al., 1999; Schaller & Valente, 2012).

$$F_{Tardy} = \sum_{j=1}^{J} \max\left(h_j + p_j - d_j, 0\right)^2$$

Lastly, we incorporate a discount (F_{seq}) in the objective, which has as goal to create room when all the buffer places at the machines are full. When the long queues at the machines fill all the buffer places, IBCs are stored on other floors, creating additional strain on shared resources. Only when the buffer is full do we give a discount in the objective for sequences of the same product. X is one if the buffer is full (see Section 4.4.1). The two requirements for giving a discount are:

1. X is 1.
2. A sequence of the same products is processed next on the machine.
    - By having the next job on the schedule be the same product as the last fixed job, $m_{s_{i,0}} = m_{s_{i,1}}$.
    - Or the next two scheduled jobs contain the same product, $m_{s_{i,1}} = m_{s_{i,2}}$.

After meeting these requirements, we add the number of IBCs of the second job in the sequence to F_{Seq}. When the following job (the third job in the sequence) also contains the same product, the number of IBCs of this job is added. This is repeated for the following jobs in the schedule and stops until a different product is in the sequence.

Figure 4.1. gives an example with five different scenarios to illustrate how F_{seq} is calculated. in Appendix IV provides a large version. We assume here that $X = 1$. The fixed jobs in the scenarios are the last two processed jobs; they cannot change. The scenarios have different fixed jobs to illustrate the workings of the calculation. In the first scenario, a product sequence forms the last fixed and the next job. The second scenario has a sequence with the two next jobs. In both cases $F_{Seq} = 2$, as we start adding the number of IBCs from the second job in the sequence. In the third scenario $F_{Seq} = 2 + 2$. In scenario 4 there is no discount to be given, as there is no sequence: $F_{Seq} = 0$. In the last scenario, the sequence does not start at the last processed or the next scheduled job; therefore $F_{Seq} = 0$.

	Scenario 1			Scenario 2			Scenario 3			Scenario 4			Scenario 5		
	Job	Product	Nr IBCs	Job	Product	Nr IBCs	Job	Product	Nr IBCs	Job	Product	Nr IBCs	Job	Product	Nr IBCs
Fixed	J-1	2323	2	J-1	2323	2	J-1	2323	2	J-1	2323	2	J-1	2323	2
	J0	8352	2	J0	5486	1	J0	8352	2	J0	8352	2	J0	5486	1
Scheduled	J1	8352	2	J1	8352	2	J1	8352	2	J4	1479	1	J2	4558	3
	J2	4558	3	J5	8352	2	J5	8352	2	J5	8352	2	J3	7335	3
	J3	7335	3	J2	4558	3	J2	4558	3	J2	4558	3	J5	8352	2
	J4	1479	1	J3	7335	3	J3	7335	3	J3	7335	3	J6	8352	2

Figure 4.1: Examples of schedules to illustrate when to give a discount for a sequence of the same product. Green indicates the jobs used in the example.

Combining objectives

Section 3.1.2 provides the following approaches for dealing with multi-criteria optimization: (1) goal programming, (2) ϵ-constraint methods, (3) lexicographic method, and (4) weighted scoring method.

Goal programming and the ϵ-constraint method require setting predefined goals, which does not apply to our problem. The lexicographic method prioritizes objectives based on their importance, which is well suited to our problem. However, this method can be computationally expensive as it requires solving the problem for each objective, which leads to a longer computation time. Lastly, the weighted scoring method allows for assigning different weights to objectives to prioritize them. However, determining the appropriate weights can be challenging.

Given these alternatives, we select the weighted scoring method to limit the computation time. In Section 5.3, we determine the weights to their appropriate values by experimenting with different configurations of the weights and analysing the individual objectives. The objective function with weights is the following:

$$\min F = w_{Makespan}F_{Makespan} + w_{Dummy}F_{Dummy} + w_{Tardy}F_{Tardy} - w_{Seq}\,F_{Seq}$$

4.6 Solution approaches

Section 3.2 states that exact or approximation methods (heuristics) can solve combinatorial optimization problems (De Giovanni, 2017). Exact methods can lead to a computationally expensive algorithm—the approximation methods usually find a feasible, not necessarily optimal, solution. However, heuristics yield a solution in a reasonable amount of time. Section 4.1 states that the problem is a real-time scheduling problem. Most existing methods for real-time scheduling use heuristics, mainly because other search and optimization methods come with less suitable, longer computation time (Ang et al., 2009).

Given these findings, we select heuristics for solving the problem. These algorithms give a suitable solution generated in real-time. In addition, when the number of jobs or machines grows at Euroma, we are not limited by a computationally expensive algorithm.

4.6.1 Constructive heuristics

This section presents four construction heuristics applicable to this problem. There is always a partial schedule known that holds the last two processed jobs. The constructive heuristics consider these jobs' colours, allergens, and certifications.

Section 3.2 shows that NNH is a relatively fast algorithm. Optimization is done more extensively by NEH but is more computationally expensive. However, it is proven to work on the similar process of scheduling the mixers at Euroma. The trade-off between the computation time and the optimisation level is interesting to research. To research this, we include both heuristics in the experiments.

The problem is solved multiple times per hour. Hence, the problems solved after each other have similar characteristics. Therefore, we propose using the last schedule constructed as the basis of the new schedule.

Recall from Section 4.1 that the set-up time depends on the certifications when using dummy jobs. We propose a heuristic that limits the dummy jobs needed to minimise this time. Because this heuristic is not derived from theory but created for this research, we explain this heuristic in more depth compared to the others.

Adapted nearest neighbour heuristic

Section 3.2 shows that the single-machine scheduling problem with sequence-dependent setup times can, in some cases, be solved as a travelling salesman problem. The set-up time includes restrictions regarding certified jobs and cleaning time. However, since the problem is limited to release dates, we adapt NNH to incorporate the release date in the start date of a job in the schedule.

The heuristic schedules the best non-scheduled job after the already scheduled jobs (recall the two fixed jobs). The best job is the job that contributes most positively to the objective. This repeats for all jobs (Oliveira & Carravilla, 2009).

Adapted NEH algorithm

The NEH algorithm sorts the jobs in descending order of processing time; the sorted list is input for the sequence. Hereafter, the jobs get a place on the schedule one by one, based on the order of the list. The place of the job is the place in the schedule that yields the best objective. This repeats until all the jobs are scheduled (Nawas et al., 1983).

Adapting the first step of NEH, defining the input sequence, impact the solution (Puka et al., 2022; Puka & Lamasz, 2022; Bhatt, 2019). Planning the hardest-to-plan jobs first yields the best results. In most problems, these are the jobs with the longest processing time. In the situation at Euroma, this could be the certified jobs. Therefore, we propose sorting the input sequence to the number of certifications, with as tie-break rule the due date. The downside to this approach is that the schedule can hold many dummy jobs after scheduling just the certified jobs. However, grouping the certified jobs would minimize the set-up time. The requirement for dummy jobs decreases by adding the suitable jobs.

Adjust the old schedule

With the constructive heuristic, adjust the old schedule (AOS), we depend on the following property: the problem to be solved is only slightly different than the problem that needed to be solved previously. Using the previous solution as the basis for the new schedule should save computational time. First, it removes the already processed jobs. The algorithm places new jobs in the queue to the place in the existing schedule, which yields the best objective, similar to NEH.

Minimal dummy jobs

Reducing the number of dummy jobs should reduce the makespan. To achieve this, we propose a constructive heuristic, minimal dummy jobs (MD), that returns a schedule with the least disruption of dummy jobs possible. Least description means first realizing the lowest number of 'stop' orders needed and second the lowest number of 'salt' orders needed; we claim minimization because it reviews all possible sequences of transitions between certifications during the scheduling process before choosing the best sequence. The remainder of this section explains the steps of this heuristic. The due date is the tie-breaker in the steps, except for the last step.

Step 1: Certifications present

Checking the job queue for jobs with certifications and identifying their certification combinations is part of the process. In the current situation at Euroma, the combinations can be kosher, halal, or both kosher and halal. A generated list outlines all possible sequences for placing these certifications. In the example in Figure 4.2 (with an enlarged version in Appendix V), kosher and halal certified jobs are present. Therefore, the list in the example is:

1. Kosher → halal
2. Halal → kosher

Step 2: Schedule jobs with certificates

Utilizing the list generated in the prior step, we plan the jobs with certificates based on the combinations specified in the list. In the provided example, for the first row in the list (kosher → halal), this means scheduling kosher jobs before halal jobs. Figure 4.2 shows this step in the example.

The jobs certified for one certification and suitable for the other (e.g., halal certified and kosher suitable) get a place after the other suitable jobs of that type. This placement gives a better starting position for the jobs following them.

Step 3: Schedule single suitable jobs

The place of jobs not suitable for one and suitable for the other certification (single suitable jobs) is in front of the certification for which they are suitable. The example in Figure 4.2 places $J1$ before $J2$.

Step 4: Schedule double suitable jobs

Jobs that are both kosher and halal suitable (double suitable jobs) can be useful for multiple certification combinations. Therefore, the algorithm checks for all present certifications in the partial schedule on how many suitable jobs they need to reach a feasible schedule without adding dummy jobs. The example shows that the kosher certified job has enough suitable jobs. The halal job, however, does still need two suitable jobs. In the example, there is only one double suitable job remaining. Because the halal certified job is the only one that needs the double suitable job, the solution is to place job $J1$ before job $J3$ in the example.

When multiple places in the schedule need double suitable jobs, the algorithm makes multiple alternative schedules that continue to the next steps.

Step 5: Add remaining jobs

The remaining jobs get behind all other jobs in increasing due date order. In addition, this step includes the addition of necessary dummy jobs. Figure 4.2 shows one necessary salt job.

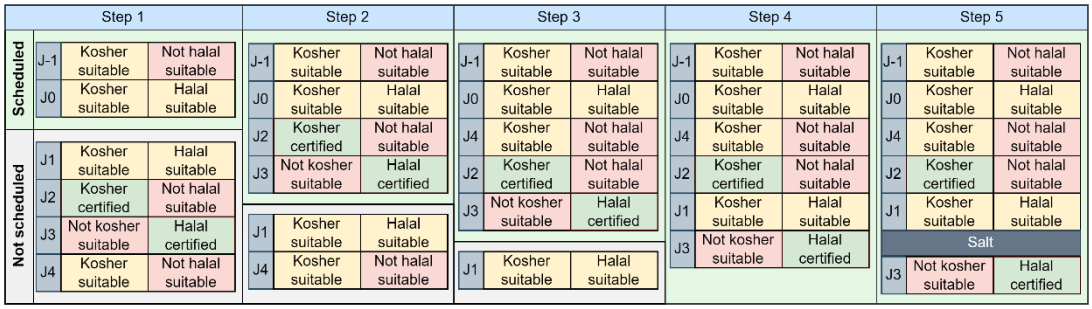


Figure 4.2: Example calculation of heuristic for minimal dummy jobs, where kosher is planned before halal.

Step 6: Compare schedules

Step 2 and Step 4 can yield multiple alternative schedules. This ensures that we always find a schedule with the least disruption of dummy jobs, as all the possible sequences with certifications are created (step 2), in addition to placing the relevant, suitable jobs before the certifications (step 2 by placing combined certified and suitable jobs after certified and non-suitable jobs, step 3, and step 4).

Figure 4.3 provides the steps in the example for the second row in the list made in step 1; scheduling halal is before kosher. Therefore, $J3$ is in front of $J2$. It shows the example for the second row on the list (see Appendix VI for a larger version). Step 4 yields two alternative schedules because $J1$ is usefull at two different places in the schedule.

We now compare the number of dummy jobs of these schedules. The schedule is judged first on the number of 'stop' jobs, then the number of 'salt' jobs, and the lowest objective function value as the last tiebreaker.

Comparing the three alternative schedules, we conclude that no schedule needs stop jobs; they all need one salt job. Therefore, we would calculate the objective and select the schedule with the best objective as the best solution.

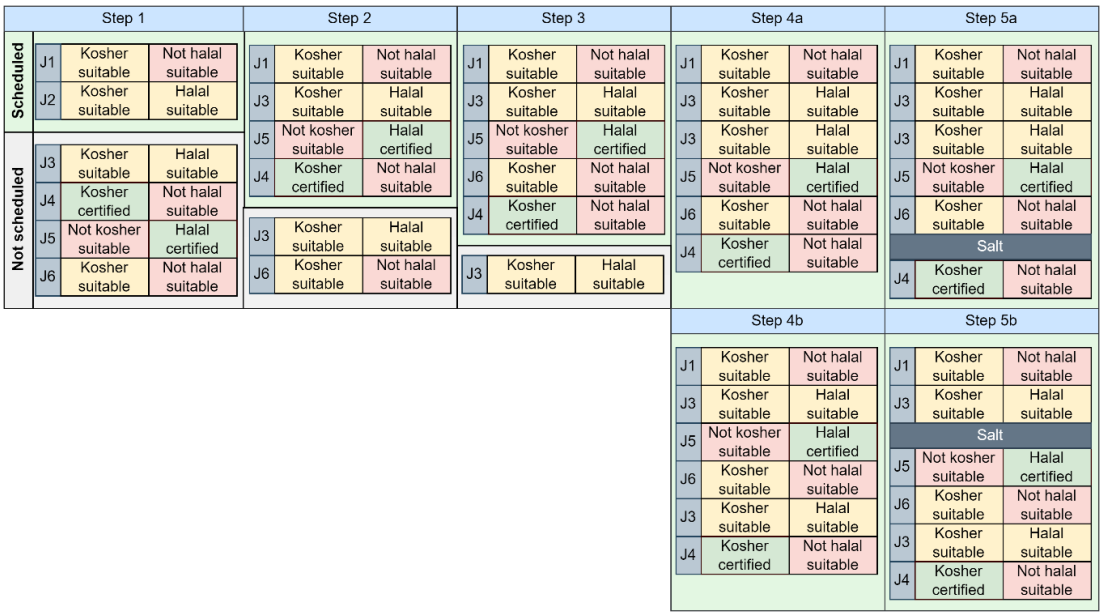


Figure 4.3: Example calculation of heuristic for minimal dummy jobs, where halal is planned before kosher.

Different sequences

The example given only included jobs with either halal or kosher certificates, not both. When there are jobs present with both certificates, next to only halal and kosher certified jobs, the list at step 1 is:

1. Kosher & halal → kosher → halal
2. Kosher & halal → halal → kosher
3. Kosher → kosher & halal → halal
4. Halal → kosher & halal → kosher
5. Kosher → halal → kosher & halal
6. Halal → halal → kosher & halal

Instinctively, it seems wise to sequence the jobs from one certificate to two certificates, back to one certificate, as in the 3rd and 4th place in the list. Alternatively, the 1st and 2nd are also good options, as after kosher & halal jobs, kosher or halal jobs do not need any additional suitable jobs. However, the 5th and 6th options also need to be investigated, as they can yield the best solution in some scenarios. Figure 4.4 (see Appendix VII for an enlarged version) gives an example of this scenario.

The example shows that the first 2 schedules need a stop job (the numbers correspond to the list with sequences), and the 3rd, 4th, and 6th need a salt job. The 5th schedule does not need any jobs and is, therefore, the best schedule.

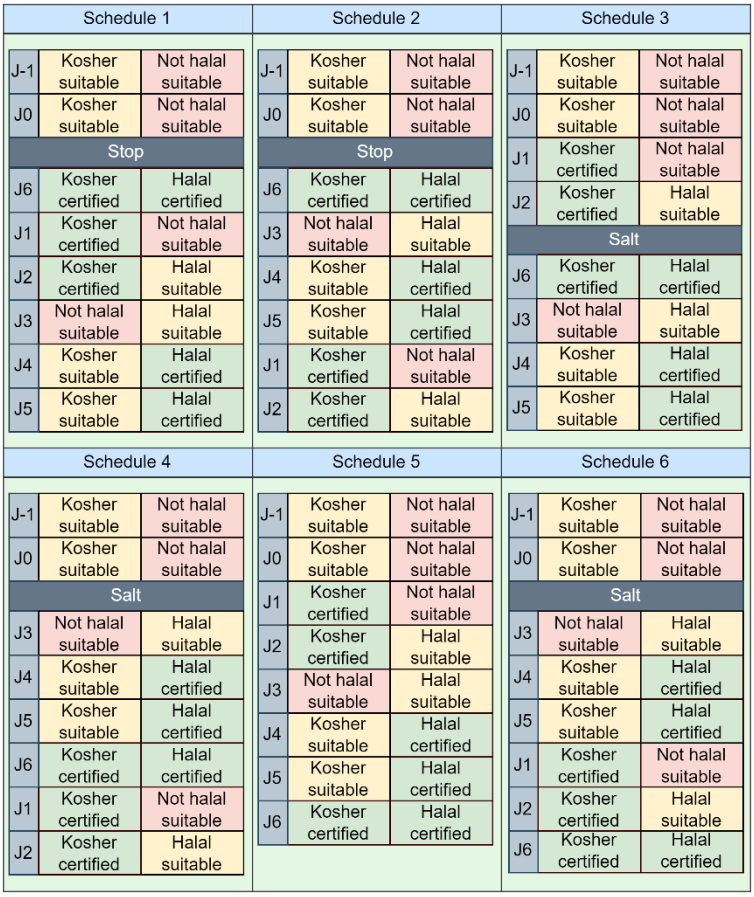


Figure 4.4: Example of possible schedules when halal, kosher, and halal & kosher certified jobs are present.

4.6.2 Improvement heuristics

Improvement heuristics SA and TS can be used to get out of a local optimum. Mexicano et al. (2023) show that for the total weighted tardiness problem with sequence-dependent set-up

time, SA was more effective than TS in finding a better solution, but TS was faster than SA. Both are desirable properties for our problem. Therefore, we test both algorithms. The literature review, Section 3.2.1, stated that SD does not come out of local optima but is very fast. It is interesting to see if the extra computational power used in SA and TS reflects in the outcome and if it outperforms the expected faster algorithm of SD.

Simulated annealing

As noted in the literature review in Section 3.2, the simulated annealing (SA) algorithm accepts a neighbouring solution S' if it yields a better or equally good objective as s. However, if S' is worse than S, SA incorporates randomness in the acceptance criterion to avoid being trapped in local optima. In such cases, the acceptance of S' is determined by a probability function that considers the difference between the objective function values of S and S', as well as the temperature T. T starts at T_0 and decreases over time with a cooling factor α. This decrease in temperature reduces the acceptance probability of worse solutions, making it less likely that SA accepts them over time. The acceptance probability function is as follows.

$$P(S, S', T) = \begin{cases} 1 & if \quad f(S') \leq f(S) \\ e^{\frac{f(S)-f(S')}{T}} & otherwise \end{cases}$$

The operator selection must ensure a connected neighbourhood (Zhang et al., 2018). We select to combine swap and move from the operators described in Section 3.2.2. Among others, Li et al. (2011) and Umam et al. (2022) use this combination in the heuristics to solve the machine scheduling problem with SA and reach a connected neighbourhood A 50/50 chance of move or swap determined the operator—the jobs or place to move to are selected randomly.

The outcomes of the experiments in Section 5.4. determine the cooling scheme, T_0, T_{max}, the Markov length, and parameter α.

Tabu search

The literature review (see Section 3.2.1) explains that TS uses a list containing (parts of) already explored solutions. This prevents exploring these solutions again and should prevent getting stuck in local optima.

The search for the best solution extends through the whole neighbourhood of a solution (in contrast to simulated annealing, which uses only one neighbour per iteration), which is computationally expensive. To limit the number of neighbouring solutions, we select only to use the swap operator. By doing so, the number of neighbourhood solutions is $n - 1$. n is the number of jobs to schedule. In addition, it leads to a connected neighbourhood. Other operators can be applied, such as move, but we exclude this for the sake of limited time.

The length of the tabu list can be static or dynamic, whereas in the latter, the size of the list changes. When the list is full, the new value replaces the first solution added. A shorter list stimulates diversification, whereas a longer list leads to intensification, where local search around the current point intensifies. We select a static list size for the simplicity of the model.

Attributes on the list can differ from changed items to entire solutions. We select as attributes the swapped jobs, meaning adding the attribute {1,2} to the list after swapping jobs 1 and 2.

Lastly, we select the number of iterations after not finding a better solution as stopping criterion. The results from experiments in Section 5.4 determine the list size and number of iterations for the stopping criterion.

Steepest Descent

In Section 3.2, we found that steepest descent searches for the best neighbouring solution until reaching a local optimum or a stopping criterion (Winston, 2004). Similar to SA, we select to use both swap and move. The algorithm ends when not finding a better solution in the neighbourhood.

4.7 Summary

In this chapter, we provide information to answer the third sub-research question:

What solutions can potentially contribute to enhancing the availability of IBCs at Euroma?

Section 4.1 provides the modelling approach; it models the problem as two separate single-machine scheduling problems to solve in real-time.

Section 4.2 lists the modelling assumptions used to set up the model description, described in Section 4.3. In the latter, we format the single-machine scheduling problem with release dates, due dates, and sequence-dependent set-up times, where the sequence-dependent set-up times depend on three consecutive jobs. Section 4.4 provides the calculation of the parameters used in the model, adapted to the processes at Euroma.

In Section 4.5, we select using the weighted scoring method to solve the multi-criteria optimization problem. Section 5.3 provides the determination of the weights.

Section 4.6 formulates alternative solutions using algorithms designed to decrease the lead time for IBCs at the packaging. Figure 4.5 illustrates the structure of this model.

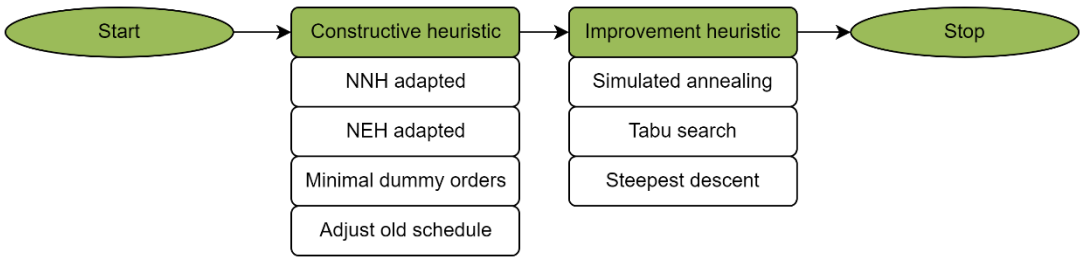


Figure 4.5: Structure of scheduling model.

The first step is to construct an initial schedule. The adapted heuristics NNH and NEH are the first two options. We develop an alternative constructive heuristic to ensure the minimal use of dummy jobs in a schedule. This heuristic is currently designed for halal and kosher certifications, as the situation at Euroma requires, but applies to any certification that looks at the last two processed activities, both in and outside the food sector. The computational cost, however, increases with the number of certifications included. Lastly, the previously made schedule forms the basis for an initial schedule.

The improvement heuristics SA and TS can overcome local optima but are computationally more extensive. SD is relatively fast but does not overcome local optima.

We explore in Chapter 5 the impact of the more computationally expensive heuristics' extra computation time on the solution compared to the cheaper options.

5 Experiments

This chapter provides information to answer the fourth sub-research question:

Which of the proposed solutions gives the best results for the situation at Euroma?

First, Section 5.1 describes the setup of the experiments. Section 5.2 gives the different experimental designs used in the later sections to find the best model configurations. Section 5.3 describes the selection of the weights needed in the objective function. Section 5.4 provides the tuning process of the SA and TS parameters. Section 5.5 provides the experiments for the different models. Section 5.6 compares the sequence made by the best model configuration to the sequence at Euroma. Section 5.7 provides insight into the impact increased demand would entail. Finally, Section 5.8 provides a summary of this chapter.

5.1 Experimental setup

This section describes the setup of the experiment. The model operates over a rolling horizon. This means that the solution to one problem impacts the next problem. The goal is to find the best solution in the long run. This means we prefer solutions that are beneficial for a longer period instead of making choices only for that moment. Therefore, the test environment for the experiments also represents this. Hence, we use a simulation that uses historical input from Euroma to test the implications of the different solutions over a longer period.

Simulation

The simulation represents a real-life situation. Therefore, this simulation does not have stochastic variables and is a deterministic simulation, where we test the impact of input parameters over a rolling horizon. This is why it might not fit the typical simulation theory framework. However, we refer to it as such in this research.

Historical input fills the queues. We select this approach rather than generating jobs because the machine input from the mixers also adheres to the cleaning and certification rules. This impacts the schedule for the packaging machines. The jobs from a mixer arrive in the queue at packaging after each other, hence the positive impact it can have on the scheduling problem. Generating these jobs would require an additional analysis of the jobs scheduled on the mixer. To simplify, we instead use historical data. We note that the queue splits for both packaging machines, but this does not entirely eliminate the correlation between jobs.

The simulation starts with a queue and the last two processed jobs per machine known, matching the real-life situation at Euroma at that point in time. The simulation includes the jobs in the queues at the moments they were available in real life.

The output of the simulation is the schedule with the best objective function values (for the Votech and the Dinnissen). Next, the simulation jumps to the next event closest in time. The events are:

- A machine becomes empty.
- A job starts in the machine.
- A new job arrives in the queue.
- A new job arrives in a mixer.
- After not creating a new schedule for 10 minutes.

Not every schedule actively contributes to selecting the next job for processing. Typically, only when a machine becomes idle, the next job selection occurs. However, all the mentioned events impact the objective regarding job scheduling, updated times, or tardiness.

Consequently, the optimal schedule can change. One of the heuristics, 'adjust old schedule' (AOS), leverages the last generated schedule to create a new one. Because of the influence of the previous schedule on the new one, we opt to employ all the listed events as triggers for generating a fresh schedule. We employ the same approach for the other heuristics to ensure a fair comparison.

At each event, the schedules of both machines are made, instead of only the machine of the job of the event. This is because the event can impact the buffer, impacting the objectives. Next, the time jumps to the next event, and the process repeats. This repeats until reaching a predetermined finishing time. For illustration purposes, when simulating one day, and 20 jobs arrive on that day, $20\ jobs * 4\ events\ per\ job = 80$ schedules, plus additional schedules when after 10 minutes no new schedule is made, are generated.

For each experiment, we simulate three non-overlapping periods of one week. Due to limited available data, we find a balance between simulated longer or multiple periods. We set the period to one week to see the impact of the model alternatives, for example, an accumulation of IBCs during the period when an algorithm performs poorly. In contrast, simulating multiple periods minimizes the input's effect on the model's performance. Section 5.2 describes any deviations from this approach. The heuristic simulated annealing (SA) depends on a random function, as described in Section 4.6. When using randomness, we include five replications per configuration to minimize the effect of randomness.

Table 5.1 shows the cleaning times of the machines based on the experience of the operators of the machines. Appendix VIII gives the contamination matrix, explaining for which colour change a dry cleaning is necessary. For the processing times of the jobs, we use the norm as used by Euroma, which depends on the quantity and the article.

Table 5.1: Cleaning times per machine.

Machine	Cleaning type	Cleaning time (min)
Votech	Dry	60
Votech	Wet	120
Dinnissen	Dry	30
Dinnissen	Wet	75

During the experiments, we encountered too positive results: the number of IBCs in the buffer went down very fast. This result can not all be credited to an algorithm, leading us to conclude that the norms used for processing and set-up times are too low. In Appendix IX, we increase the norms for a more realistic situation. The experiments done in this chapter use the adjusted norms.

Output parameters

The results of the simulations are compared to each other using multiple parameters. The objective represents the combined optimization criteria. In addition, we measure the time it takes to construct the schedules, which we refer to as CPU time, to indicate the configuration's suitability to a real-time setting. The maximum CPU time allowed is 1 second to act fast in the factory setting. We run the experiments using Delphi 11 on a computer with an Intel Core 1.00GHz processor and 12.5GB of RAM.

Per experiment, additional parameters can be useful, for example, looking at individual objective function values for determining the weights for the objective or the number of dummy jobs needed. When this is the case, we refer to it in Section 5.2.

When we compare objective function values of multiple simulated periods, we use the average objective weighted over time, with as time the time before a new schedule is made. We use a weighted objective method to prevent a high impact of very similar problems to the average objective.

5.2 Experimental design

This section outlines the experimental design for each experiment, while the subsequent sections present the corresponding experiment outcomes. As the results reveal unexplored yet promising areas, the experimentation scope broadens to encompass various input parameters in those sections.

Recall that in Section 4.5 we stated that the number of jobs to be penalized in the objective function for needing a dummy job is r. Due to limited time, we do not experiment with this number and set the number of jobs to penalize on 3. We select this number because when these jobs are processed, we expect new suitable jobs to be added to the queue before the tardiness becomes too much of a problem. This number could be experimented with, e.g., setting it higher, lower, or dependent on the queue length.

Experiment 1: Objective weight selection

Section 4.5 described that the objective includes weights to balance the individual objectives. For this experiment, we select using the NNH heuristic and steepest descent. Both heuristics are not dependent on randomness, which gives the experiments an equal ground. We select the best configurations by comparing the individual objectives, in addition to the time spent on dummy jobs and the average number of IBCs. This shows the impact of penalizing dummy jobs in the objective and the flow of the IBCs.

As described in Section 4.5, the makespan is the most important objective to increase the availability of the IBCs. Therefore, we set this weight to at least 50% for each experiment, dividing the remaining fraction over the other three objectives. Each weight has a minimum of 10% and increases with steps of 10% to create impact in each experiment. Given these bounds, we experiment with every configuration of the weights possible.

Experiment 2: Simulated annealing parameters

For simulated annealing, the parameters T_o, T_{max}, Markov length and α need to be established. We first determine T_o and T_{max}, by looking at the acceptance ratio per temperature (the fraction of proposed worse solutions accepted compared to the current solution) at the end of each Markov chain. Using preliminary experiments, we find the range in which T_o and T_{max} should be between 0.0001 and 20. Subsequently, these values are input into the experiment to determine T_o and T_{max}. As initial parameters, we set the Markov chain length to 500 and define α = 0.99. We run the test for 10 individual problems, all from different days (not using simulation), and take the averages over the results found.

Next, we determine the Markov chain lengths (250, 500, 750) and values for α (0.98, 0.99, and 0.995). The experiments include all combinations of the parameters, as they both heavily impact the exploration and exploitation dynamics of the algorithm. We compare the results by the difference in objective function values and the CPU time. The experiment includes all configurations for the Markov chain length and value of α. NNH is the constructive heuristic.

Experiment 3: Tabu list length and stopping criterion

The third experiment determines the tabu list length and stopping criterion. Section 4.6.2 describes that the tabu list length is static. However, we expect that the queue length impacts

the optimal list length. For example, the impact of the list length is different if the queue holds 100 jobs or 10 jobs. Therefore, during the experiments, the list length changes to fit the number of items in the queue. The possible number of possible exchanges (which is the attribute on the tabu list) is $NrPE = \frac{J(J-1)}{2}$. In this experiment, we test a percentage of this number as length.

A total of 5 values of percentages are experimented with, covering a range from 10% to 60% with steps of 10%. We opt to conclude at 60% due to the anticipation that a higher value would excessively constrain possible swaps.

The heuristic stops when not finding a better solution after a preset number of iterations. In this experiment, we test different numbers of iterations, namely 10, 25, 50, and 100. We select the value which leads to the lowest objective function value within 1 second.

We use 100 iterations after a better solution as stopping criterion for the experiments for selecting the tabu list length. The tabu list length selected in this experiment determines the value used in the stopping criterion selection. NNH is the constructive heuristic.

Experiment 4: Heuristics
This experiment tests the performance of different heuristics. As constructive heuristics, we test adapted NNH, adapted NEH, adjusting the old schedule (AOS), and minimal dummy jobs (MD), as described in Section 4.6.1. The experiments combines every one of these constructive heuristics to each improvement heuristic: Simulated annealing (SA), Tabu search (TS), and steepest descent (SD), as well as a benchmark without an improvement heuristic. The performance analysis compares the average objective function values and the CPU time. The best configuration is the configuration which gives the lowest objective with the average CPU time below 1 second.

5.3 Determining weights

Experiment 1: Objective weight selection
Table 5.2 provides the results of the experiments for weight selection. It shows the averages of each objective, the number of IBCs, and the time spent on dummy jobs. The highlighted cells represent the best results per objective. Recall that the objectives are $F_{makespan}$, F_{tardy}, F_{dummy}, and F_{seq}, which give the total makespan of a schedule, the tardiness squared, a penalty for scheduling dummy jobs at the beginning of the schedule, and a discount for sequences of the same product when the buffer is full, respectively.

Configuration 6 yields the lowest value for $F_{makespan}$ and F_{tardy} at the Votech machine, where 7 is the second-best option for $F_{makespan}$ and a close third for F_{tardy}. It is noteworthy that the value of w_{tardy} is relatively small in this configuration. This can be expected, as the value of F_{tardy} is high in comparison with the other objectives. Consequently, a lower value achieves a better balance in this regard.

For the Dinnissen machine configuration, number 6 is in the top 20% for all objectives except F_{seq}. However, for this machine, we see that the values are very low: a $F_{makespan}$ of around 7 hours, and F_{tardy} of a fraction of the tardiness at the Votech. The small queue at the Dinnissen in comparison to the Votech can explain this. The latter is the biggest bottleneck of this process and the most improvements we expect here. Therefore, we look at only the best configurations for the Votech for this and further experiments.

Table 5.2: Experiment 1, results: individual objectives for weight selection

	Weight				Votech				Dinnissen					
Nr	$w_{makespan}$	w_{tardy}	w_{dummy}	w_{seq}	$F_{makespan}$	F_{tardy}	F_{dummy}	F_{seq}	$F_{makespan}$	F_{tardy}	F_{dummy}	F_{seq}	**IBCs**	**Dummy time (h)**
1	0.5	0.3	0.1	0.1	1.52	7.14	0.003	0.004	0.29	0	0	0.002	15.9	9.33
2	0.5	0.2	0.1	0.2	1.49	7.5	0.002	0.035	0.27	0.004	0	0.009	16.1	17.33
3	0.5	0.2	0.2	0.1	1.49	7.5	0.002	0.035	0.27	0.004	0	0.009	16.1	16.33
4	0.5	0.1	0.2	0.2	1.38	7.77	0	0.042	0.26	0.037	0	0.005	15.8	0
5	0.5	0.1	0.3	0.1	1.48	7.77	0	0.042	0.26	0.037	0	0.005	15.8	0
6	0.5	0.1	0.1	0.3	1.38	7.14	0	0.004	0.26	0	0	0.002	15.9	0
7	0.6	0.2	0.1	0.1	1.4	7.16	0	0.004	0.29	0	0	0.003	16.3	0
8	0.6	0.1	0.2	0.1	1.48	9.09	0.003	0.065	0.29	0.005	0	0.014	17.4	0
9	0.6	0.1	0.1	0.2	1.48	9.07	0	0.065	0.29	0.005	0	0.014	17.4	0
10	0.7	0.1	0.1	0.1	1.49	7.48	0.003	0.037	0.29	0.004	0	0.014	16.5	1.00

The value of F_{dummy} of the Votech is very low, indicating that there are not many dummy jobs planned. The last column of the table, dummy time, gives the total hours needed for dummy jobs in the 3 weeks of the experiments. Recall that a stop job equals 24 hours, and a salt job is 0.5 hours. The results confirm that the schedules do not use many dummy jobs. In addition, not always when a penalty occurs, a dummy job processes (dummy time is zero, while the penalty is more than zero). This indicates that the dummy job stood before the machine's second or third next job, and suitable jobs arrived in the queue before the job processes. This aligns with expectations, as the penalty increases if the job is first in the schedule.

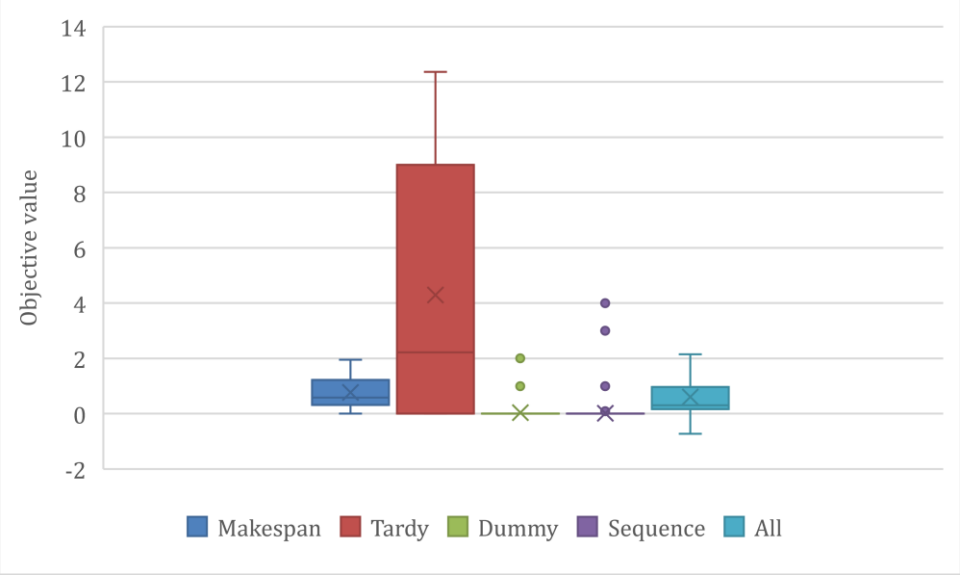


Figure 5.1: Distribution of objective values.

Overall, the best configuration found is from configuration 6, where the values of $w_{makespan}$, w_{tardy},w_{dummy}, and w_{seq} is 0.5, 0.1, 0.1, and 0.3, respectively. However, the first three values include the lower bound set for the weight, suggesting the presence of potentially better, unexplored configurations. Figure 5.1 shows the distribution as a boxplot of the objective values using the sixth configuration.

The objective function value for tardiness stands out because it spans over a considerable interval with high outliers. This is to be expected, as it represents the squared values. The penalty for the dummy jobs and the discount for a sequence is primarily 0 and has few outliers. The makespan has no apparent outliers and heavily affects the overall objective, which is logical as it represents 50% of the objective.

With this information, we explore additional configurations that span beyond the original set bounds of the weights but are in the area in the best found configuration. $w_{makespan}$ still represents the most important goal; hence, the new interval is still a substantial portion [0.4-0.55]. The values for tardiness are substantially higher than for the other objectives. Therefore, the new interval of w_{tardy} is [0.025-0.1.25]. The intervals for w_{dummy} and w_{seq} are [0.05-0.15] and [0.25-0.35] respectively. Every possible configuration of the weights adding up to 1 is tested, with intervals of 0.025.
Table *5.3* shows the two best configurations for each objective of the Votech $F_{penalty}$ as this is often 0. Appendix X gives the complete results.

Table 5.3: Additional experiments weight selection.

Weight				Votech					
$w_{makespan}$	w_{tardy}	w_{dummy}	w_{seq}	$F_{makespan}$	F_{tardy}	F_{dummy}	F_{seq}	IBCs	Dummy time (h)
0.425	0.125	0.1	0.35	1.50	7.13	0.000	0.007	16.4	0.09
0.425	0.125	0.125	0.325	1.50	7.13	0.000	0.007	16.4	0.09
0.45	0.075	0.125	0.35	1.48	9.07	0.000	0.065	17.4	0.00
0.45	0.075	0.15	0.325	1.48	9.09	0.000	0.065	17.4	0.00
0.5	0.075	0.075	0.35	1.36	7.16	0.000	0.004	15.5	0.00
0.5	0.075	0.1	0.325	1.36	7.16	0.000	0.004	15.5	0.00
0.5	0.075	0.125	0.3	1.36	7.16	0.000	0.004	15.5	0.00
0.5	0.075	0.15	0.275	1.36	7.16	0.000	0.004	15.5	0.00

The bottom four rows, the first two rows and the third and fourth rows give the same result over all parameters, indicating the generation of equal or similar schedules. Noteworthy is that this shows that the configuration of $w_{makespan}$ and w_{tardy} seem to make the most impact as w_{dummy} and w_{seq} differ within these groups. Because not all combinations of w_{dummy} and w_{seq} are covered in the groups, we conclude they do impact the schedule.

The four bottom rows give a minimal value regarding $F_{makespan}$. The tardiness of these configurations is close to the lowest value of F_{tardy} found, 7.16 and 7.13 respectively. In addition, it uses no dummy jobs. The discount represented by F_{seq} is low in these configurations; this can indicate a low number of IBCs in the buffer, consistent with the high number of IBCs at the highest value of F_{seq} in the third and fourth rows.

We conclude that the best configuration is in the four bottom rows. Of these, we select the configuration 0.5, 0.075, 0.125, and 0.3 to use in the following experiments, where w_{dummy} and w_{seq} are middle values of the four configurations.

5.4 Determine parameters heuristics

5.4.1 Simulated annealing

Experiment 2.1: Simulated annealing – Start and lower temperature

The experiment for determining the starting temperature includes comparing the acceptance ratio and the temperature. Figure 5.2 displays the average acceptance ratio per temperature found in the experiments. It shows that the acceptance ratio gradually decreases.

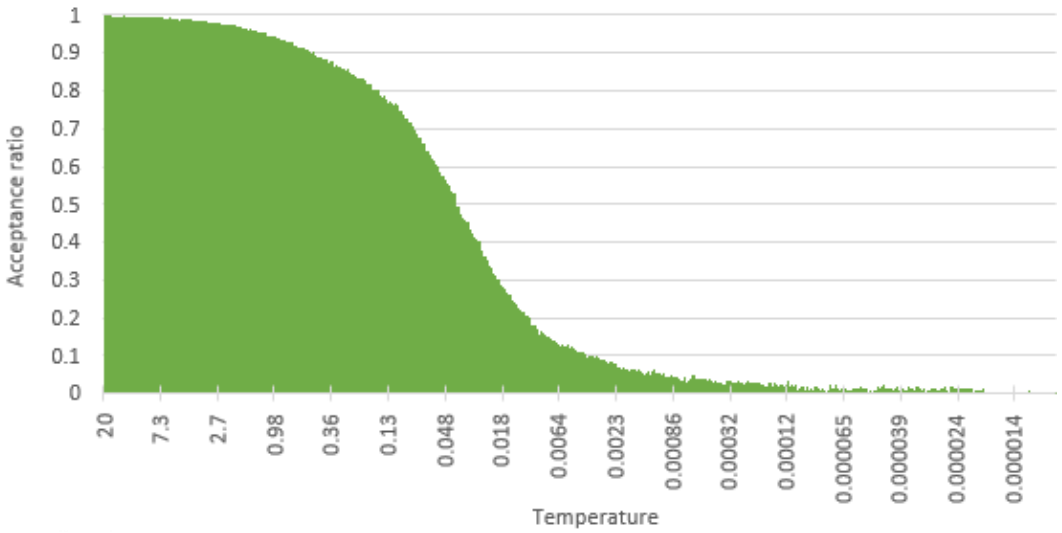


Figure 5.2: Experiment 2.1, results: temperature and acceptance ratio.

When the temperature is high, the acceptance ratio is close to 1. There, almost all solutions are accepted. Towards the end, this goes down, and only better solutions are accepted. If the temperature is between 20 and 0.13, we see that the chance of acceptance is above 75%. We select to set the starting temperature at 0.13, where the acceptance ratio is around 75%. When starting at this temperature, we still have a high acceptance ratio at the beginning without spending computational time exploring with an even higher acceptance ratio. We accept that we might get stuck in a local optimum by not accepting every selected neighbour. However, since the acceptance ratio is still high at the start, we find this possibility acceptable.

When the temperature is low, the acceptance ratio approaches zero. We expected this, as the probability of selecting a worse is also near zero. The acceptance rate is close to zero around the temperature of 0.0001. After this point, we do not expect to find new and better solutions. In conclusion, we select a starting temperature of 0.13 and a lower bound of 0.0001. We test this configuration in the remaining experiments.

Experiment 2.2: Simulated annealing – Markov chain length and value of α

To evaluate the different Markov chain lengths and values for α, we compare the objective function value and the CPU time. As explained in Section 5.3, we only focus on the settings for the Votech. Figure 5.3 shows the results.

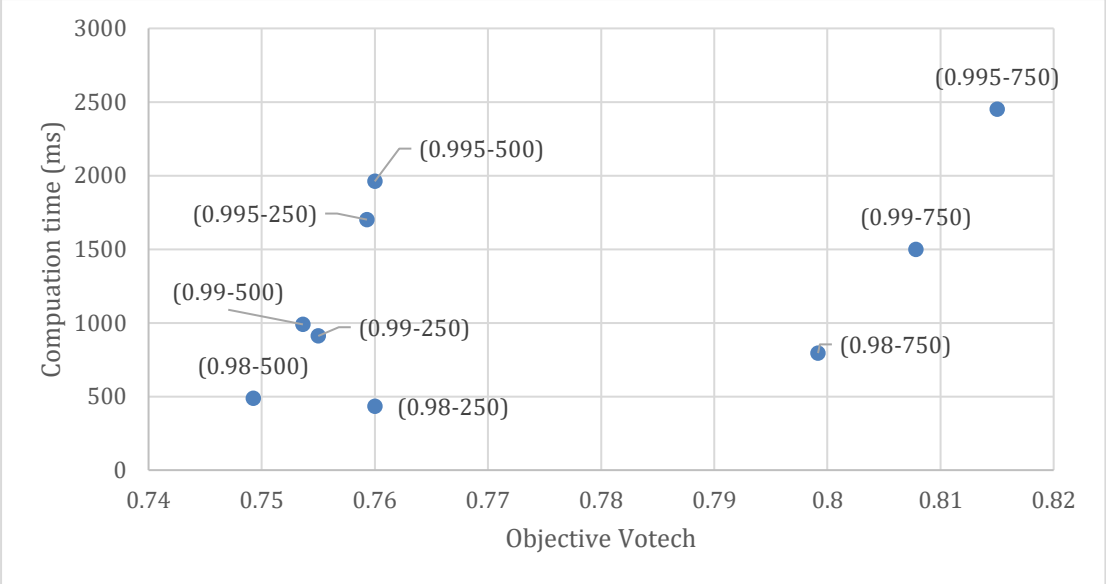


Figure 5.3: Experiment 2.2 results: objective Votech and computation time.

First, analysing the computation time, we see results logical results: the configuration with the highest value of α and the Markov chain length has the highest CPU time, and the lowest values of the parameters yield the lowest CPU time. Higher values of the Markov Chain yield more iterations per temperature, and the higher value of α means that the temperature decreases in smaller steps, resulting in more iterations and therefore a CPU time rise.

Next, we focus on the objective. Surprisingly, the lowest parameters yield the lower objective function values and vice versa. This is surprising because we would expect that more iterations would lead to a higher probability of finding a good solution. However, it can also mean that more bad solutions are accepted when the temperature is high for longer. This can lead to a bad final solution.

The results show that for the best objective value, the best values for α and the Markov chain length are 0.98 and 500, respectively. The CPU time is around half a second, therefore well below the set threshold of 1 second. Therefore, we choose this as the configuration to use in the heuristic.

5.4.2 Tabu Search

Experiment 3.1: Tabu search – Tabu list length

The results of experiment 3.1 show the impact of the length of the tabu list. Figure 5.4 displays the results. The objective is the lowest when the tabu list holds 40% of the possible solutions. Recall from Section 3.2.1 that the implications of a too long or too short tabu list need a compromise between limiting the choices of moves in a too long tabu list without premature convergence in a smaller list. We find this balance at 40% of the potential swaps. We choose not to explore more in the range of around 40% as this may cause overfitting. We use this parameter for the list length for all the following experiments.

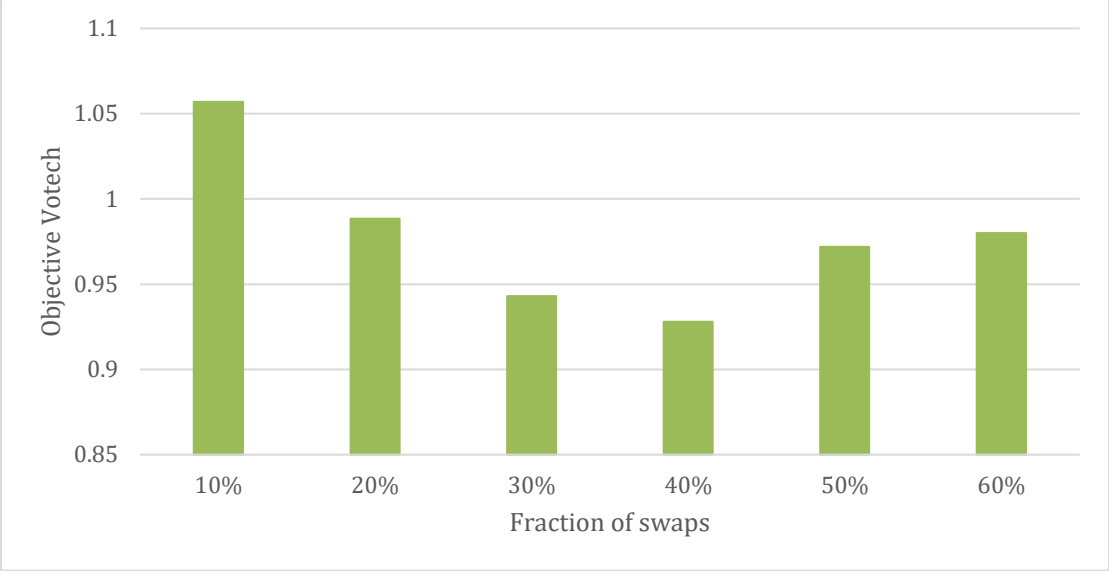


Figure 5.4: Experiment 3.1, results, objective Votech and the fraction of possible swaps.

Experiment 3.2: Tabu search – Stopping criterion

The goal of this experiment is to select after what number of iterations not finding a better solution the algorithm stops. Figure 5.5 gives the results of the stopping criteria selection. It shows a high objective when the number of iterations is small. This is to be expected, as it invests no time in exploring. From 40 iterations onwards, the extra iterations stop leading to improvements in the schedule, and the objective stays the same. The CPU time is for all configurations below the one-second threshold. We select 60 as our parameter. This allows the algorithm more iterations should the problem instances become more complex, but without adding strain to the computation time.

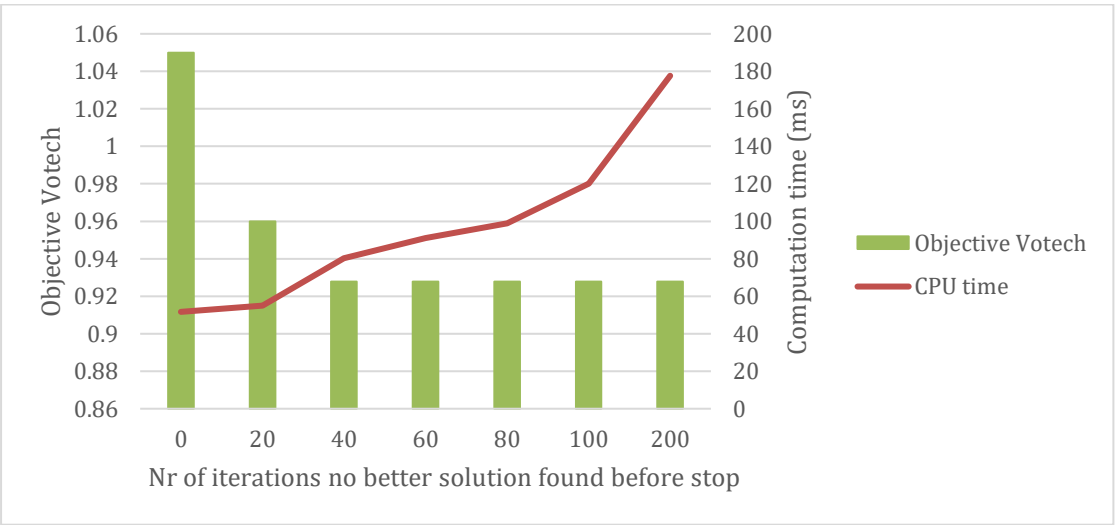


Figure 5.5: Experiment 3.2, results: objective Votech, computation time and nr of iterations no better solution found.

5.5 Model alternatives

Experiment 4: Heuristics

In the fourth experiment, we compare the performance of the different constructive and improvement heuristics. Table 5.4 provides the results of the experiment. The colours indicate a good (green) or bad result (red) for the objective and CPU time for the combination of constructive and improvement heuristics.

Table 5.4: Experiment 4, results: objectives of Votech and CPU time per configuration.

	Objective Votech				CPU time (ms)			
Heuristic	**SA**	**SD**	**TS**	**No IH**	**SA**	**SD**	**TS**	**No IH**
AOS	0.75	0.81	0.81	2.03	688	65	71	39
MD	0.77	1.07	0.96	4.80	653	85	99	36
NEH	0.75	0.80	0.81	0.85	636	73	78	43
NNH	0.81	0.94	0.91	2.12	645	71	77	42

First, we analyse the experiments without an improvement heuristic (column 'No IH'). The differences in performance are big. The objective of MD is very high. This is logical, as finding a low objective is not the primary concern of this heuristic. NEH performs best, followed by AOS, which is not surprising, as AOS uses NEH to schedule new jobs. The CPU time is between 36 and 43 milliseconds and, therefore, negligible. Interestingly, from the theory, we expected that NEH would be more time extensive (Section 3.2.1). However, the results do not confirm this.

The outcomes with the improvement heuristic show that all the solutions generated by only the constructive heuristics can still be improved. SA gives the best results, and SD and TS are comparable. Notably, SD gives better results than TS when using NEH as a constructive heuristic. This is possible because the tabu list saves swaps instead of whole solutions. In addition, the constructive heuristic, NEH, already gives a great solution and the diversification of TS improves less than the intensification of SD. This also explains that the performance is best at SA, where diversification and intensification are combined.

The combination of constructive and improvement heuristics yields a comparable ranking to only using a constructive heuristic. NEH scores better than AOS, MD, and NNH. NNH performs poorly, which can be because the initial schedule is in a local optimum. This explanation is affirmed by the relatively good results when combining NNH with SA. The combination of NEH or AOS and SA gives the best objective values. We do note that MD and SA give results close to these configurations.

Regarding the CPU time, the algorithms show a big difference. However, the solution forms below a second; hence, we do not exclude any solutions for this reason.

In the previous experiments, we excluded the Dinnissen machine from the selection of parameters. However, the algorithm still needs to work for this machine. Therefore, we also display the impact of the heuristics on this machine in Table 5.5. The table shows that the objectives are still very low, and the differences between all the objectives are minimal. Therefore, all configurations of constructive and improvement heuristics are acceptable when applied to the Dinnissen machine.

Table 5.5: Experiment 4, results: objectives of Dinnissen per configuration

	Objective Dinnissen			
Heuristics	**SA**	**SD**	**TS**	**No IH**
AOS	0.11	0.123	0.114	0.124
MD	0.112	0.11	0.113	0.114
NEH	0.11	0.113	0.117	0.121
NNH	0.114	0.117	0.125	0.127

Given these findings, we conclude the following. AOS and NEH give the best results of the constructive heuristics. Of the improvement heuristics, SA works best. The combination of NEH and AOS results in the lowest objective for the Votech, closely followed by NEH and SA. Regarding computation time, there are no notifiable differences between the constructive heuristics. However, SA takes significantly longer compared to the other improvement heuristics. Because this is still within bounds, we accept this. Therefore, we choose configurations AOS and SA as well as NEH and SA to use in the following experiment.

5.6 Comparison with the current situation

The configurations found as best in Section 5.5, NEH with SA and NEH with SA, are compared with the sequence of jobs that happened at Euroma - without using an algorithm. We compare the same 26 consecutive days. Due to the different processing and set-up times, we cannot reasonably compare objectives. Instead, we compare the number of processed jobs, cleanings needed, and dummy jobs needed.

We recognize that the situations cannot be compared precisely for multiple reasons. First, despite using the same days, with the same starting situation and input, there is still the possibility that processed jobs differ during the 26 days. This can result in more cleaning or dummy jobs in one of the situations. However, as 26 days is a relatively long period, we expect this to have a minimal effect. In contrast, having a small queue, caused by faster processing of the jobs, there are fewer jobs to choose from, which can result in a schedule that needs more cleaning than when having a longer queue. Concurrently, a smaller problem instance - occurring when the queue is small - can be solved more easily and may result in a solution closer to the optimum. Lastly, the processing times and set-up times used in the simulation are based on an estimation, while the real-life sequence is not. While recognizing this, we still think a comparison is valuable.

Table 5.6 shows the results, it uses V for the Votech and D for the Dinnissen. The upper rows show the number of transitions of certifications where dummy jobs are necessary. The real-life situation needs 7 ‘stop’ jobs and 3 ‘salt’. The scenario created using NEH, uses 1 salt job in 1 of the 5 simulations. AOS uses no dummy jobs. This shows that the algorithm works well regarding preventing transitions to certified jobs without 2 suitable jobs.

Table 5.6: Results of comparing the current situation and the model. Period of 26 days.

Heuristics	**Current**			**NEH+SA**			**AOS+SA**		
Certification transition	**Halal**	**Kosher**		**Halal**	**Kosher**		**Halal**	**Kosher**	
Nr. ‘stop’ needed	4	3		0	0		0	0	
Nr. ‘salt’ needed	1	2		0.2	0.2		0	0	
	V	**D**	**Total**	**V**	**D**	**Total**	**V**	**D**	**Total**
Nr. wet cleaning	87	64	151	46	59	105	47	59	106
Nr. dry cleaning	28	18	46	23	23	46	24	22	46
Nr. cleaning total	115	82	197	69	82	151	71	81	152
Nr. jobs processed	297	271	568	309	272	581	308	275	583

The number of cleanings at the Votech decreases significantly. It reduces from 197 to around 152 when using an algorithm. The total number of cleanings at the Dinnissen does not change using the algorithm. However, the increased number of dry cleanings decreases the number of wet cleanings for both machines. This is preferable as dry cleaning takes less time than wet cleaning. Similar to what we found in earlier experiments, the improvement at the Dinnissen machine is limited, which is explained by having fewer jobs in the queue for the machine,

leading to fewer jobs to choose from to create an optimal schedule. By consolidating all cleaning activities and converting them to hours per day, the model effectively decreases the daily cleaning time from 17.0 to 11.5 (NEH) and 11.6 hours (AOS) for both machines.

Lastly, the table shows the number of jobs processed. This increases, but only by 2.5%. The demand is the same, so processing many more jobs is impossible. Hence, the number of jobs processed indicates that there is room for processing more jobs, but with the provided demand for this experiment, we cannot indicate how many jobs this can be.

The previously listed results show a very positive impact. However, the number of tardy jobs increased. Table 5.7 shows the number and percentage of jobs per tardiness interval (for example, 19 jobs were between 0.5 and 1 day too late at Euroma). Recall that a job becomes tardy when it stands in the buffer for 24 hours. The results show that most jobs do not become tardy (interval [0-0] in Table 5.7). This percentage is higher in the current situation. However, after 0.5 days tardy, the algorithms perform better than the current situation. This suggests that the algorithm successfully reduces the occurrence of highly tardy jobs but at the expense of an increased number of jobs experiencing tardiness. Section 6.2 provides suggestions for improvements in this area.

Table 5.7: Results, number of jobs per tardiness interval (per 12 hours). The highlighted percentages represent the highest values.

Interval tardy (days)	Number of jobs in the interval			Aggregated percentage		
	Current	NEH	AOS	Current	NEH	AOS
[0-0]	472	461	450	83.1%	79.3%	78.6%
<0-0.5]	45	75	78	91.0%	92.3%	91.9%
<0.5-1]	19	25	28	94.4%	96.6%	96.4%
<1-1.5]	14	10	12	96.8%	98.3%	98.1%
<1.5-2]	5	5	6	97.7%	99.1%	99.0%
<2-2.5]	4	3	1	98.4%	99.7%	99.1%
<2.5-3]	2	1	8	98.8%	99.8%	99.8%
<3-3.5]	4	0	0	99.5%	99.8%	99.8%
<3.5-4]	0	1	1	99.5%	100.0%	100.0%
<4-4+>	3	0	0	100.0%	100.0%	100.0%

5.7 Increased demand

The last section described that the demand limits the number of processed jobs. Since Euroma aims to grow in the long run, it is useful to explore how much demand the machines can handle. To address this, we conduct experiments involving increased demand scenarios to assess their effects.

Appendix XI outlines the methodology for increasing demand. We experiment over the same two weeks. We choose this extended timeframe instead of multiple smaller ones to gain insight into potential job accumulation when demand exceeds machine capacity. The results of Section 5.5 and Section 5.6 show minimal differences in performance between the NEH and AOS heuristics. We arbitrarily select NEH as a constructive heuristic to limit experimental runtime in this experiment. Table 5.8 summarizes the outcomes.

Table 5.8: Objectives, the average number of IBCs in buffer, time spent on dummy jobs, and CPU time per demand increase.

	Votech				Dinnissen						
Increased demand	$F_{makespan}$	F_{tardy}	F_{dummy}	F_{seq}	$F_{makespan}$	F_{tardy}	F_{dummy}	F_{seq}	IBCs	Dummy time (h)	CPU (ms)
0%	1.12	4.2	0.03	0.11	0.21	0.06	0.00	0.00	11.4	0.00	630
5%	1.76	10.3	0.03	0.00	0.06	0.11	0.00	0.06	14.8	0.00	642
10%	1.80	15.4	0.01	0.24	0.11	0.21	0.01	0.04	17.5	0.00	660
15%	2.46	26.4	0.04	0.62	0.41	0.11	0.00	0.19	23.8	0.00	676
20%	2.94	40.4	0.02	0.70	0.48	0.21	0.00	0.57	30.3	0.00	696
25%	3.50	61.4	0.04	0.91	0.33	1.03	0.00	0.87	39.5	0.00	712
30%	3.98	78.0	0.02	1.21	0.50	1.47	0.00	0.72	47.6	1.00	730

The results show that the objective values all grow except F_{dummy}, which stays around 0. This is to be expected, as when more jobs are available, the number of suitable jobs also rises. Comparing the number of dummy jobs processed in the simulation, the schedule contains either zero dummy jobs or one salt job in the two weeks. Therefore, we state using dummy jobs is still limited with increased demand.

The growth of $F_{makespan}$ is logical because by scheduling more jobs, it would take longer to process them all. Figure 5.6 shows the number of jobs and cleaning jobs processed. The growth in processed jobs aligns with expectations, while the count of cleaning operations shows a decline. Considering allergen and colour factors, we attribute this to the algorithm's improved sequencing under increased demand when having a larger pool of jobs in the queue to select from.

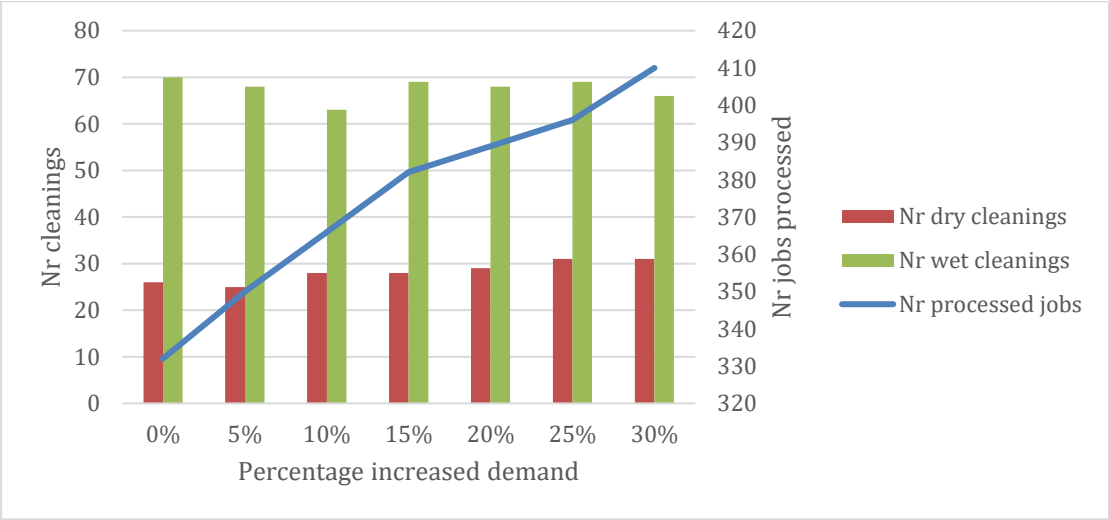


Figure 5.6: Production and cleaning jobs processed with increased demand.

Table 5.8 highlights F_{tardy}, representing squared tardiness, while Figure 5.7 presents a graphical representation of tardiness in processed jobs (not squared) for easier comparison. The depicted figure demonstrates that the maximum tardy job duration expands to 15 days at a 30% increase, exceeding the simulated time of 14 days. We note that this can occur when a job was already in the queue for at least a day prior to simulation commencement. Although average tardiness demonstrates a more gradual growth, the rate of increase remains steep.

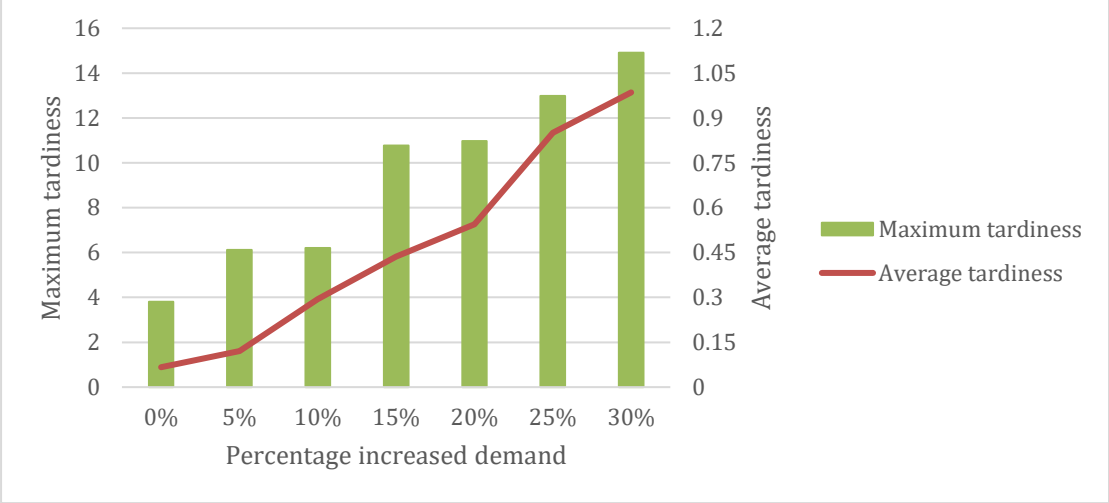


Figure 5.7: Maximum and average tardiness with increased demand.

The increase of F_{seq} is a predictable outcome concurrent with the growth in the number of jobs, thus IBC volumes. Figure 5.8 displays the average number of IBCs per day. Demand increase of 0%, 5%, and 10% shows a gradual decline in the number of IBCs present. At 0%, the buffer has an average reduction of around 10 IBCs. We credit the decline to the improvements in the algorithm compared to the real-life situation at Euroma. A 20% demand increase yields a stable IBC count range, around the average in the buffer in the current situation. However, demand increments of 25% and beyond lead to IBC accumulation, signalling the machine's incapacity to manage heightened demand loads.

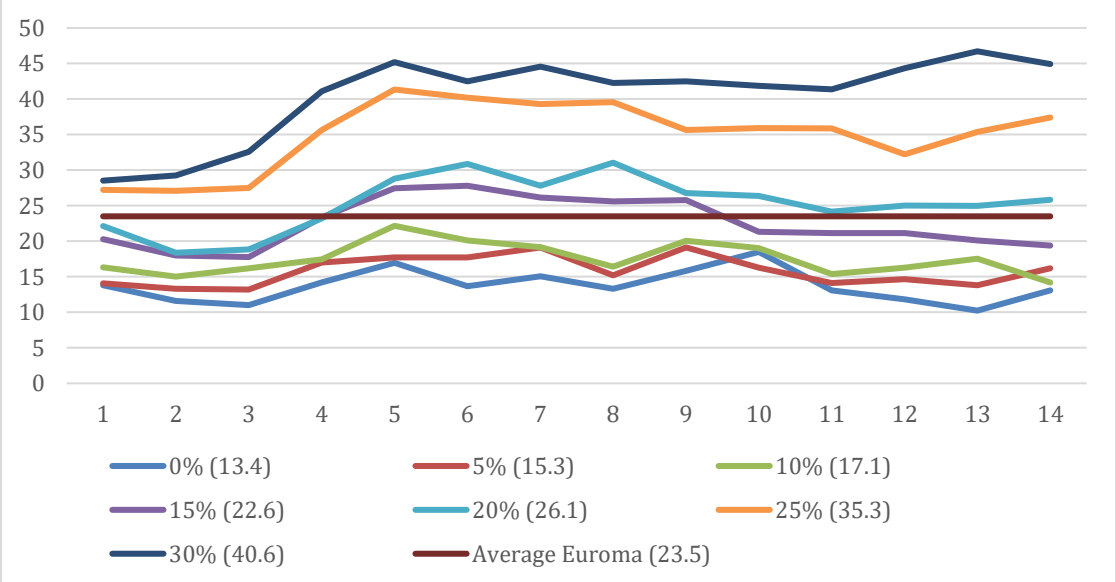


Figure 5.8: IBCs in buffer over time with increased demand. With () indicating the average number of IBCs per increased demand.

Lastly, we note the rise in CPU time in the last column of Table 5.8. As job volume increases, problem complexity follows suit. The observed increase in computation time is thus a foreseeable outcome. Notably, the algorithm retains computational efficiency within the predefined one-second threshold. However, it is prudent to acknowledge the potential for threshold breach as complexity grows, necessitating adjustments to the algorithm for sustained performance.

5.8 Summary

This chapter provides information to answer the fourth sub-research question:

Which of the proposed solutions gives the best results for the situation at Euroma?

Section 5.1 gives the experimental setup. Because the purpose of the model, working in real-time in a factory setting, we use a simulation to test it. The input is the historical data of Euroma. The first results show that the norm times are too low for processing and set-up time. To be able to conduct the experiments, the norms are adjusted.

Section 5.2 gives the experimental design. This holds the different experiments conducted. It includes the objective weight selection, SA parameters, tabu list parameters, and model alternatives.

Section 5.3 presents the weight selection for the objective. Primarily, it finds a balance between the impact of the makespan and tardiness on the schedule.

Section 5.4 provides the selection of parameters for SA and TS. An analysis of the acceptance ratio determines the starting and lower bound temperature. Markov chain length and the value of α undergo testing together, and we compare the results by examining the objective and CPU time. In the case of TS, experiments with different percentages of potential exchanges set the tabu list length. A comparison between the objective function values and CPU times of different configurations determines the stopping criterion.

Section 5.5 gives the experiments regarding the model alternatives. This entails experimenting with all constructive and improvement heuristics. We conclude that of the constructive heuristics, AOS and NEH both in combination with SA, perform best. Regarding computation time, the combined heuristics are all below the set upper bound of 1 second.

Section 5.6 compares the sequence of the best model alternatives with the sequence that happened at Euroma. The sequence made by the algorithm performs better regarding the number of dummy jobs and cleaning jobs. The percentage of extremely tardy jobs also decreases. However, the percentage of jobs which become tardy is higher than at Euroma. Lastly, the number of jobs processed shows a slight improvement. Therefore, we test in Section 5.7 the effects of increased demand.

Sections 5.5 and 5.6 neither show a significant superiority in performance between the combination of NEH with SA and AOS with NEH. Therefore, both combinations are suitable to use.

The increased demand in Section 5.7 shows an immediate growth in tardiness and makespan when the demand increases. However, the number of cleaning jobs decreased due to better sequencing. The number of IBCs in the buffer throughout the simulation shows that the machines can handle the demand until a 20% increase in jobs. In addition, at 20% or below, the strain on IBCs is not higher than in the current situation at Euroma.

Finally, it is important to highlight that these results rely on adapted time parameters and simulation data. More accurate parameters should match the conditions at Euroma, improving the accuracy. Furthermore, practical testing is necessary to solidify the model's viability in real-world scenarios. However, the results show promising improvements regarding IBC use, cleaning time, and dummy jobs.

6 Implementation

This chapter advises on how to implement the model at Euroma by answering the fifth research question:

How can the solution be implemented at Euroma?

Section 6.1 describes the tool made. Section 6.2 gives an overview of how to integrate the tool into the current systems. Lastly, Section 6.3 advises on how to implement the tool in practice.

6.1 Tool

A tool uses the model developed in this research. This tool gathers and transposes information from the systems (see Section 6.2 for more information on the systems) to feed the model. In addition, the tool transposes the schedule generated by the model to fit within the systems.

The tool runs every time one of the following events occurs:

- A machine becomes empty.
- A job starts in the machine.
- A new job arrives in the queue.
- A new job arrives in a mixer.
- After not creating a new schedule for 10 minutes.

Upon establishing a new schedule, a series of events occur. The tool does not execute the events but sets them in motion (see Section 6.2 for more details). The following events are triggered:

- The schedule is updated.
- The AGVs are triggered to move the IBCs for a job whose status becomes ‘active’.

Table 6.1 (see Appendix XII for an enlarged version) shows an example of a schedule an operator sees. The schedule indicates to the operator what to expect regarding cleanings and product changes. Each schedule is also saved in an archive to reflect and improve the model later.

Table 6.1: Example of a schedule.

Machine	Job	Article	Article Description	Time in Mixer	Time in buffer	Status	Planned time start	Planned time stop	NrIBCs	Color	Halal certification	Kosher certification	Allergens
Z410	J00058025	59883	Jus Rundvlees	9-06-23 17:51	9-06-23 20:09	Stopped	9-06-23 23:50	10-06-23 1:14	2	Bruin	Halal suitable	Trefa	
Z410	J00058189	51108	Oosterse melange	9-06-23 1:41	9-06-23 3:18	Active	10-06-23 1:38	10-06-23 2:55	2	Lichtbruin	Halal suitable	Kosher suitable	
Z410	J00057271	65893	Oosterse mix	9-06-23 3:38	9-06-23 5:26	Released	10-06-23 3:11	10-06-23 4:43	2	Lichtbruin	Halal suitable	Kosher suitable	
Z410	J00058024	59417	BBQ classic	9-06-23 16:49	9-06-23 18:56	Released	10-06-23 5:07	10-06-23 6:51	2	Oranje	Halal	Kosher	
Z410	J00058449	60931	Fajita mix	10-06-23 1:10		Mixer	10-06-23 7:15	10-06-23 8:51	2	Rood	Halal suitable	Kosher suitable	
Z410	J00058432	52426	Bolognese	9-06-23 6:15	9-06-23 7:57	Released	10-06-23 9:15	10-06-23 11:11	3	Oranje	Halal suitable	Kosher suitable	So;So
Z410	S00000001	67106	Dry Cleaning				10-06-23 11:11	10-06-23 12:47					
Z410	J00058415	65139	Curry Japans	9-06-23 16:35	9-06-23 17:31	Released	10-06-23 12:47	10-06-23 14:28	2	Groen	Haram	Trefa	Gl;Me;So;So
Z410	J00058250	55158	Curry Geel	10-06-23 0:57		Mixer	10-06-23 14:52	10-06-23 16:01	2	Donkergeel	Halal suitable	Kosher suitable	Gl;Me;So;So
Z420	J00058420	66204	Champignons saus	9-06-23 16:25	9-06-23 18:04	Stopped	9-06-23 23:57	10-06-23 0:58	1	Geel	Halal suitable	Trefa	Mo
Z420	J00058418	66204	Champignons saus	9-06-23 20:33	9-06-23 23:02	Active	10-06-23 0:58	10-06-23 1:59	1	Geel	Halal suitable	Trefa	Mo
Z420	J00058245	65745	Nacho mix	9-06-23 19:54	9-06-23 20:47	Released	10-06-23 6:04	10-06-23 6:43	1	Geel	Halal suitable	Kosher suitable	Mo
Z420	J00058244	65745	Nacho mix	9-06-23 16:31	9-06-23 18:52	Released	10-06-23 6:43	10-06-23 7:22	1	Geel	Halal suitable	Kosher suitable	Mo
Z420	S00000002	67107	Wet Cleaning				10-06-23 7:22	10-06-23 10:34					
Z420	J00058525	61775	Nasi kruiden	9-06-23 18:06	9-06-23 19:42	Released	10-06-23 10:34	10-06-23 11:16	1	Geel	Halal	Kosher suitable	So
Z420	J00058526	61775	Nasi kruiden	10-06-23 1:03	10-06-23 2:45	Mixer	10-06-23 11:16	10-06-23 11:58	1	Geel	Halal	Kosher suitable	So

6.2 Implementation architecture

The tool needs to be integrated with the systems present at Euroma (see Section). Figure 6.1 shows how the tool interacts with the systems.

Production jobs from ESA include details of the start and stop times. MES provides information about the last two processed jobs, including dummy jobs and the most recent cleaning jobs. The purpose of extracting the cleaning job is to establish a distinct job number for subsequent cleaning processes. This approach ensures proper documentation of cleaning jobs for quality

assurance purposes. LN contains information about article colours, certifications, and the contamination matrix (see Section 5.1). Additionally, PLS Pro contains allergen information per article.

Data extraction from ESA and MES occurs every 5 minutes to provide real-time insight into the ongoing production status and to identify any significant events as described in Section 6.1 promptly.

Every 12 hours, information from LN and PLS Pro updates. This schedule aligns with the infrequent nature of changes in this data. For example, changes in allergen composition require modifications in various departments, such as product development, quality control, and planning. Consequently, these changes require more than 12 hours to propagate and impact the packaging stage of the product.

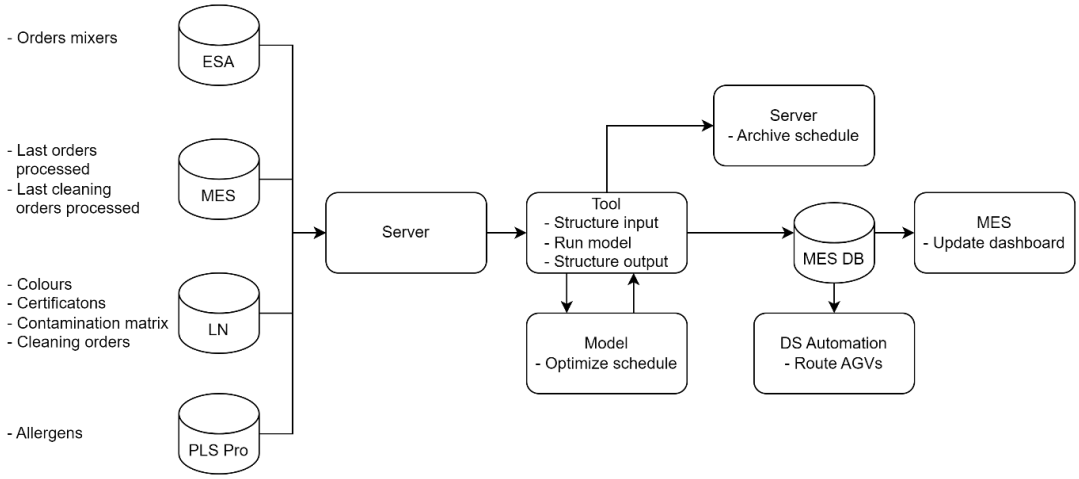


Figure 6.1: Interaction between tool and current systems

The server stores the acquired data. Subsequently, the tool processes this data for utilization within the model, undertaking calculations as detailed in Section 4.4. Once the model finalizes the scheduling, the tool arranges the schedule's format to facilitate storage within the MES database. MES updates the dashboard on the production floor, ensuring that operators remain informed in addition to calling AGVs to transport IBCs for the impending job.

The server holds an archive containing the schedules and an error log. The error log serves as a repository for tool activities, including data extraction and schedule creation, as well as any identified discrepancies, such as absent data or incomplete scheduling. The preservation of the schedules and error log is to reflect and improve the scheduling process.

6.3 Implementing the tool in practice

An engineer leads the implementation and requires close cooperation with a diverse team, including operators, team leaders, planners, and the ICT department. The engineer expertise should encompass knowledge of the process at Euroma and proficiency in programming and scheduling.

Presently, the determination of the next processed job lies with the operators. The tool supersedes this approach by presenting the operators with a schedule-based job processing sequence. While we have conducted testing and comparative analysis of the algorithm in alignment with the scenario outlined in Section 5.6, we recognize disparities that prevent an entirely equitable comparison. Therefore, we recommend a phased approach to tool

implementation, initially focusing on providing schedule recommendations. This enables operators to assess potential tool deficiencies, propose enhancements, and exercise discretion in following the suggestions.

Instances with disregarded scheduling suggestions are documented and reflected on by the engineer and operators. Proposed improvements undergo testing within a simulated environment before being integrated into the tool, provided they demonstrate improved outcomes.

Table 6.2 outlines the proposed implementation timeline. The programming of the tool is based on the model developed in this research but requires adaptation to suit the company's environment. Concurrently, adjustments need to be made to the current systems to accommodate this tool. Once the technical implementation is complete, testing occurs within the factory environment. This testing phase, known as 'shadow running,' allows operators and planners to provide feedback on the suggested schedules. Weekly evaluations of the findings will inform ongoing adjustments to the tool. While initially set at four weeks, extending this period is crucial if improvements are still necessary. This extension is particularly important because the final step involves full integration, requiring operators to strictly adhere to the schedule without personal adjustments when they deem necessary.

Table 6.2: Timeline implementation.

	Week						
Activity	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Programming tool							
Adjusting current systems							
Shadow running							
Full integration							

6.4 Summary

The chapter discusses the implementation of a model at Euroma in response to the fifth research question:

How can the solution be implemented at Euroma?

Section 6.1 integrates the developed model into a practical tool that collects and transforms information from various systems to feed the model. The tool operates based on specific events such as machine status changes, job start, and new job arrivals. It triggers events upon establishing new schedules, updating the schedule and prompting AGVs to move IBCs for active jobs.

Section 6.2 focuses on the implementation architecture. The tool must integrate with existing systems at Euroma. Data extraction takes place every 5 minutes or 12 hours, based on the changing nature of the data. Acquired data is transformed in the tool and used as input for the model, and the output is formatted and stored in the MES database.

Section 6.3 outlines the practical implementation of the tool, involving an engineer, operators, team leaders, unit managers, and the ICT department. The tool replaces the manual job sequencing operators do with a schedule-based approach. We propose a phased implementation strategy, starting with providing schedule recommendations to operators for assessment and improvement proposals. Instances of disregarded recommendations serve as

evaluation points for potential improvements. An evaluation in a simulated environment first tests the proposed before being incorporated into the tool.

Overall, the chapter guides the integration of the model into a practical tool, its interaction with existing systems, and its gradual implementation in collaboration with various stakeholders at Euroma.

7 Conclusion and recommendations

This chapter concludes the thesis by first answering the main research question in Section 7.1. Next, Section 7.2 gives recommendations to the company. Section 7.3 provides the discussion and limitations of this research. Section 7.4 discusses the contribution of this research to the literature and practice. Lastly, Section 7.5 outlines potential avenues for future research to further enrich and expand upon the current findings.

7.1 Conclusion

Euroma's strategic goal is to increase its output. A critical bottleneck in production is the availability of IBCs within the mixing department. This situation is compounded by the accumulation of filled IBCs in the packaging process, stagnating the release of new jobs. The corresponding research question is:

How can the lead time at the packaging process be decreased to improve the availability of IBCs at *Euroma?*

Our analysis concludes that we can achieve the most improvements by reducing the extended waiting time at the packaging process, thereby enhancing the throughput time of the IBCs. In line with this finding, we define the main objective of the research as finding a solution that contributes to decreasing the lead time at packaging to enhance the availability of the IBCs.

We model the problem as two single-machine scheduling problems solvable in real-time, with additional constraints for product sequencing. We propose combinations of 4 constructive and 3 improvement heuristics as solution approaches. The constructive heuristics include a developed heuristics aimed to minimize the number of dummy jobs necessary, MD. The objectives include four individual objectives combined with the weighted scoring method.

The experiments include testing constructive and improvement heuristics combinations in a simulated environment. The combination of AOS and SA and NEH and SA gives the best results for minimizing the objective. The CPU times are within set bounds. MD minimized the dummy jobs scheduled. However, the other constrictive with improvement heuristics performed almost equally well regarding minimising dummy jobs. MD with SA generated results regarding the objective close to the best two configurations.

We compare the two best found configurations to the situation at Euroma. Table 7.1 summarizes the differences between the situation at Euroma and the simulation. The percentage gives $\frac{Current-Algorithm}{Current} * 100\%$ and is green or red when it is a positive or negative outcome, respectively.

The results of SOA or NEH with SA do not differ significantly. The algorithms yield better results than the current situation, especially regarding the number of dummy jobs and cleanings needed. It reduces the cleaning time from 17.0 hours per day to 11.5 (NEH) and 11.6 hours (AOS). However, the total amount of tardy jobs increased, although the amount of extremely tardy jobs decreased.

Table 7.1: Overview of all results of the algorithms compared to the current situation at Euroma. Given in percentage and absolute change from the current situation.

	Current			NEH + SA			AOS + SA		
Certification transition	**Halal**	**Kosher**		**Halal**	**Kosher**		**Halal**	**Kosher**	
Nr. 'stop' needed	4	3		0 (-100%)	0 (-100%)		0 (-100%)	0 (-100%)	
Nr. 'salt' needed	1	2		0.2 (-80%)	0.2 (-90%)		0 (-100%)	0 (-100%)	
	V	**D**	**Total**	**V**	**D**	**Total**	**V**	**D**	**Total**
Nr. wet cleaning	87	64	151	46 (-47%)	59 (-8%)	105 (-30%)	47 (-46%)	59 (-8%)	106 (-30%)
Nr. dry cleaning	28	18	46	23 (-18%)	23 (28%)	46 (0%)	24 (-14%)	22 (22%)	46 (0%)
Nr. cleaning total	115	82	197	69 (-40%)	82 (0%)	151 (-23%)	71 (-38%)	81 (-1%)	152 (-23%)
Nr. jobs processed	297	271	568	309 (4%)	272 (0%)	581 (2%)	308 (4%)	275 (1%)	583 (3%)
Percentage jobs tardy			83.1%			79.3% (-5%)			78.6% (-5%)
Percentage jobs > 1 day tardy			94.4%			96.6% (2%)			0.964 (2%)
Percentage jobs > 2 days tardy			97.7%			99.1% (1%)			99.0% (1%)

Experiments with increased demand, in line with Euroma's objective to increase production, show a rise in both tardiness and makespan. However, the implementation of improved sequencing results in a decrease in the number of cleaning jobs. The analysis of IBC quantities indicates that the machines can effectively handle demand growth of up to 20% in jobs. Beyond this point, the quantity of IBCs at the buffer surpasses the buffer and machine capacity. This surplus would have a detrimental impact on the overall availability of IBCs, the primary focus of this research.

Compared to the current situation, our proposed model exhibits significant improvements across almost every KPI at various demand levels, except for the count of tardy jobs. In Section 7.2, we provide recommendations for addressing this issue. Overall, the reduction in the required number of IBCs, a contribution aligned with Euroma's primary goal, drops from 23.5 in the current situation to 13.4. Additionally, when using our model, the demand can be elevated by up to 20% before the buffer reaches the same occupancy level as observed in the current scenario.

7.2 Recommendations

We recommend adopting a real-time scheduling model designed to improve job sequencing using NEH or AOS in conjunction with SA optimization techniques. Among the various configurations considered, these approaches demonstrate the most favourable performance concerning the defined objective while falling within acceptable CPU time limits. We first recommend testing both models in practice before selecting one as superior.

The algorithm outperformed real-world scenarios across all dimensions except for the count of tardy jobs. We propose improving this aspect by refining the objective's weighting scheme or redefining the tardiness criteria. Moreover, the current determination of job tardiness, set at 24 hours after buffer arrival, does not apply to all jobs. To rectify this, we recommend

introducing a due date specific to each article or article group. However, implementing such article-specific due dates requires data collection for each article.

We recommend integrating mixer planning with machine operations to enhance the current setup. The current proposed model focuses solely on optimizing scheduling within the packaging machines based on the demand stemming from the mixers. To improve overall efficiency, it is beneficial to synchronize the mixer planning to ensure the processing of suitable jobs in the mixer process before certified jobs that require the same packaging machine. This also applies to allergens and colour changes. A harmonized coordination has the potential to increase operational efficiency significantly.

7.3 Discussion and limitations

In this model, we integrated all constraints related to tardiness and certifications as soft constraints incorporated in the objective function. The outcomes of our analysis demonstrate the effective handling of certifications through this approach. However, the prevention of tardiness does not yield the desired results. In retrospect, exploring the incorporation of tardiness as a hard constraint could offer potential advantages in preventing tardiness, albeit at the expense of potential increases in dummy jobs, cleaning jobs, and makespan. Since the MD heuristic yields the minimum number of dummy orders, we can use the calculation employed in this heuristic to establish a constraint concerning the minimal use of dummy jobs.

Our findings indicate that under conditions of low demand, the sequencing effectiveness diminishes due to the limited selection of jobs. While we tested the model under regular and heightened demand scenarios, we have not examined scenarios with reduced demand (e.g., holidays) or machine disruptions (e.g., maintenance). As a result, the model's optimal performance might not extend to these situations.

Currently, the processing times for jobs have fixed values. In reality, such times exhibit variability. We choose to exclude this variability from the model for simplicity. However, in hindsight, incorporating this variability would likely yield more accurate results, provided accurate data is available.

Due to time constraints, we did not conduct experiments varying the value of r, which represents the number of jobs integrated into the penalty within the objective function for dummy job requirements. We fixed this value at 3. There is potential for improvements by experimenting with different values, such as increasing, decreasing, or making it dependent on the queue size. However, since the best algorithms found already minimized the use of dummy jobs, we do not anticipate significant improvements in this aspect.

It is important to acknowledge that data availability and reliability limited the research. While sequence data for machines and mixers were accessible, the recorded times were sometimes unrealistic. We observed instances where, theoretically, one machine started processing a new job while still processing another job, or where a job seemingly took negative time to process. Consequently, we heavily leaned on established norms for time values rather than historical data, which also proved unreliable but was adjusted to be more realistic.

7.4 Contribution

Our examination of the existing literature highlighted an absence of addressing sequence-dependent setup times for sequences of more than two tasks. To address this gap, we created a model that considers the two most previously completed jobs and the upcoming jobs in the

sequence. This approach bridges the knowledge gap in effectively handling setup times within extended task sequences.

Our research contributes to scheduling optimization, focusing on the company's operational context. By integrating real-time scheduling techniques with the company's data, we harnessed practical insights to increase efficiency. Additionally, our work extends the understanding of job-dependent setup times by considering sequences of three tasks, offering a more accurate representation of real-world scenarios.

We introduced an approach that merges existing heuristics with penalty mechanisms to prevent using dummy jobs in scheduling. This approach not only streamlines scheduling but also improves overall robustness. Acknowledging the practicality of dummy jobs, we incorporated their presence into our model while maintaining optimized, feasible schedules. Our contributions bridge theoretical advancements and pragmatic implementation, empowering our company with improved scheduling techniques that reflect real-world complexities.

We developed a heuristic approach aimed at minimizing the necessity for dummy jobs. While the outcomes of this research reveal that the NEH and AOS combined with SA strategies also reduce the requirement for dummy jobs, they do not guarantee the absolute minimum. Our algorithm can offer significant value in situations where the utmost reduction of dummy jobs is paramount.

The model is designed with the company in mind, ensuring seamless integration into existing operations. Moreover, the generality of the model makes it applicable to various organizations, requiring only input modifications to suit different contexts. This approach is especially useful when real-time information impacts the schedule and is impacted highly by sequence-dependent set-up time.

7.5 Future research

The first suggestion for future research involves exploring the potential benefits of integrating other departments into the scheduling process. This extension could encompass not only the mixers but also the precursor and subsequent steps in the operational sequence, such as replenishment and packaging for consumers. We can uncover opportunities to develop more comprehensive and harmonized scheduling strategies by thoroughly examining how these inter-departmental interactions impact scheduling dynamics.

Another significant aspect for future investigation revolves around refining the scheduling model to minimize tardiness. This involves researching the underlying factors contributing to delayed job completion and identifying critical parameters within the model. Strategies to mitigate tardiness may involve optimising task sequences, assigning varying priority levels, or incorporating dynamic adjustments to account for real-time delays.

In a dynamically changing operational landscape, assessing the impact of fluctuating demand patterns, such as those experienced during holidays, is imperative. Furthermore, examining the consequences of machine breakdowns and unplanned disruptions on scheduling outcomes is necessary. We can develop robust strategies to navigate unforeseen interruptions by simulating different scenarios and analysing their effects while maintaining optimal scheduling performance.

Another research line entails investigating how variable task times influence the scheduling model. This exploration involves introducing variety in processing and setup times. Analyzing the implications of this can fine-tune the model to accommodate better real-world variations. This adaptation would ultimately lead to creating scheduling strategies that are more accurate and reliable in practical scenarios.

In essence, these envisioned paths of future research hold the potential to extend and refine the current scheduling at Euroma and other organizations. By venturing into the integration of departments, tackling tardiness, addressing demand fluctuations and disruptions, and accommodating variable task times, the future of scheduling optimization poises to deliver more resilient, adaptable, and practical strategies in meeting the demands of complex operational environments.

Bibliography

Ahmad, M. (2023, February 27). *First-come, first-served (FCFS) scheduling algorithm*. Retrieved from Educative.io: https://www.educative.io/answers/first-come-first-served-fcfs-scheduling-algorithm

Anderson, E., Glass, C., & Potts, C. (2003). *Local Search in Combinatorial Optimization.* Princeton: Princeton University Press. doi:https://doi.org/10.1515/9780691187563-014

Ang, A. T., Sivakumar, A. I., & Qi, C. (2009). Criteria selection and analysis for single machine dynamic on-line scheduling with multiple objectives and sequence-dependent setups. *Computers & Industrial Engineering, 56*(4), 1223-1231. doi:https://doi.org/10.1016/j.cie.2008.07.018

Arora, J. S. (2017). *Introduction to Optimum Design* (4th ed.). Academic Press.

Barnes, J. W., & Vanston, L. K. (1981). Scheduling Jobs with Linear Delay Penalties and Sequence Dependent Setup Costs. *Operations Research, 29*(1), 146-160. doi:https://doi.org/10.1287/opre.29.1.146

Benthem, v. T. (2021). *Solving a multi-objective hybrid flow shop.* University of Twente.

Bhatt, P. (2019). *Permutation Flow Shop via Simulated Annealing and NEH.* UNLV Theses, Dissertations, Professional Papers, and Capstones. doi:http://dx.doi.org/10.34917/15778402

Červeňanská, Z., Važan, P., Juhás, M., & Juhásová, B. (2021). Multi-Criteria Optimization in Operations Scheduling Applying Selected Priority Rules. *Applied Sciences, 11*(6). doi:https://doi.org/10.3390/app11062783

Cinar, D., Topcu, Y., & Oliveira, J. (2015). A Taxonomy for the Flexible Job Shop Scheduling Problem. *Optimization, Control, and Applications in the Information Age*, 17-37.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduciton to Algorithms* (3th ed.). Cambridge: The MIT Press.

De Giovanni, L. (2017). *Methods and Models for Combinatorial Optimization: Heuristis for Combinatorial Optimization.* Retrieved from Depertimento Matematica: https://www.math.unipd.it/~luigi/courses/metmodoc1819/m02.meta.en.partial01.pdf

De Souza, E. A., Nagano, M. S., & Rolim, A. G. (2022). Dynamic Programming algorithms and their applications in machine scheduling: A review. *Expert Systems with Applications*. doi:https://doi.org/10.1016/j.eswa.2021.116180

Eardley, N. (2014, May 12). *What is halal meat?* Retrieved from BBC.com: https://www.bbc.com/news/uk-27324224

Euroma. (2019). *2019: Euroma on the move*. Retrieved from Euroma: https://www.euroma.com/en/innovation/products

Euroma. (2022). *History*. Retrieved from Euroma: https://www.euroma.com/en/about/Over-Euroma

Glover, F. (1989). Tabu Search—Part I. *ORSA Journal on Computing, 1*(3), 190-206. doi:https://doi.org/10.1287/ijoc.1.3.190

Graham, R., Lawler, E., Lenstra, J., & Rinnooy Kan, A. (1979). Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey. *Annals of Discrete Mathematics, 5*, 287-326. doi:https://doi.org/10.1016/S0167-5060(08)70356-X

Haupt, R. (1989). A survey of priority rule-based scheduling. *OR Spektrum*(11), 3-16.

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems.* The MIT Press.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. Science. *Science, 220*(4598), 671-680. doi:DOI: 10.1126/science.220.4598.671

Kosher Certifications & Supervisions. (2022). *Kosher Certifications & Supervisions*. Retrieved from Kosher : https://www.kosher-kcs.com/wat-is-kosher/

Lalla, E. A. (2022). Lecture 3. Supplier selection and multi-objective. Enschede.

Lee, K., Leung, J., & Pinedo, M. (2010). Makespan minimization in online scheduling with machine eligibility. *4OR - A Quarterly Journal of Operations Research*, 331-364. doi:https://doi.org/10.1007/s10288-010-0149-1

Li, K., Yang, S., & Ma, H. (2011). A simulated annealing approach to minimize the maximum lateness on uniform parallel machines. *Mathematical and Computer Modelling, 53*, 854-861.

Lin, S., & Kernighan, B. (1973). n Effective Heuristic Algorithm for the Traveling-Salesman. *Operations Research, 21*(2), 498-516. doi:https://doi.org/10.1287/opre.21.2.498

Mexicano, A., Carmona-Frausto, J., Montes-Dorantes, P., Cervantes, S., Cervantes, J., & Rodríguez, R. (2023). Problem, Simulated Annealing and Tabu Search for Solving the Single Machine Scheduling. *Lecture Notes in Networks and Systems* (pp. 86-95). Springer International Publishing.

Miettinen, K. (2004). *NONLINEAR MULTIOBJECTIVE.* New York: Springer Science+Business Media, LLC.

Minetti, G., Alfonso, H., & Gallard, R. (2001). *Solving the single machine scheduling problem with sequence-dependent set-up times as a TSP via evolutionary algorithms.* Retrieved from Academia.edu: https://www.academia.edu/77601257/Solving_the_single_machine_scheduling_problem_with_sequence_dependent_set_up_times_as_a_TSP_via_evolutionary_algorithms

Mustu, S., & Eran, T. (2018). The single machine scheduling problem with sequence-dependent setup times and a learning effect on processing times. *Applied Soft Computing*, 291-306.

Nagano, M. S., & Miyata, H. H. (2016). Review and classification of constructive heuristics mechanisms. *The International Journal of Advanced Manufacturing Technology, 86*, 2161-2174. doi:https://doi.org/10.1007/s00170-015-8209-5

Nawaz, M., Enscore, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega, 11*(1), 91-95. doi:https://doi.org/10.1016/0305-0483(83)90088-9

Oliveira, J. F., & Carravilla, M. A. (2009). *Heuristics and Local Search.* Retrieved from Feculdade de Engenharia da Universidade do Porto: https://paginas.fe.up.pt/~mac/ensino/docs/OR/CombinatorialOptimizationHeuristicsLocalSearch.pdf

Pinedo, M. (2016). *Scheduling: Theory, Algirthms and Systems.* (5th, Ed.) New York: Springer.

Puka, R., & Lamasz, B. (2022). Impact of input sequence on N-NEH+ algorithm. *Proceedings 10th Carpathial Logistics Congress*, (pp. 191-196).

Puka, R., Duda, J., & Stawony, A. (2022). Input Sequence of Jobs on NEH Algorithm for Permutation. *Management and Production Engineering Review*, 32-43.

Radar, D. J. (2010). *Deterministic Operations Research: Models and methods in linear optimization.* Terre Haute: John Wiley & Sons, Inc.

Ramalhinho-Lourenço, H. (2019). *Metaheuristics for Combinatorial Optimization.* Retrieved from Univerisitat Popeu Fabra: https://www.upf.edu/documents/3186295/0/Combinatorial+Optimization+Problems/0bbbfc98-dbc9-2e38-802b-499b9347ad1b

Reddy, R. (2019). Combinatorial Optimization Using (Integer) Linear. *Journal of Contemporary Issues in Business and Government, 25*(01), 150-163. doi:10.47750/cibg.2019.25.01.009

Rosing, K., ReVelle, C., Rolland, E., Schilling, D., & Current, J. (1998). Heuristic concentration and Tabu search: A head to head comparison. *European Journal of Operational Research*, 93-99.

Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research, 177*(3), 2033-2217. doi:https://doi.org/10.1016/j.ejor.2005.12.009

Ruiz, R., & Vázquez-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem. *European Journal of Operational Research, 205*(1), 1-18. doi:10.1016/j.ejor.2009.09.024

Salhi, S. (2002). Defining tabu list size and aspiration criterion within tabu search methods. *Computers & Operations Research, 29*, pp. 67-86.

Schaller, J., & Valente, J. (2012). Minimizing the weighted sum of squared tardiness on a single machine. *Computers & Operations Research*, 919-928.

Sun, L., Cheng, X., & Liang, Y. (2010). Solving Job Shop Scheduling Problem Using Genetic Algorithm with Penalty Function. *International Journal of Intelligent Information Processing, 1*(2), 65-77. doi:10.4156/ijiip.vol1.issue2.7

Sun, X., Noble, J., & Klein, C. (1999). Single-machine scheduling with sequence dependent setup to minimize total weighted squared tardiness. *IIE Transactions*(31), pp. 113-124.

Tahami, H., & Fakhraver, H. (2022). A Literature Review On Combining Heuristics and Exact. *European Journal of Information Technologies and Computer Science*, pp. 6-12.

Umam, M., Mustafid, M., & Suryono, S. (2022). A hybrid genetic algorithm and tabu search for minimizing makespan in flow shop scheduling problem. *Journal of King Saud University - Computer and Information Sciences*, 7459-7467.

Vollmann, T. E., Berry, T. E., & Whybark, D. C. (1992). *Manufactering planning and control systems* (3nd ed.). United States of America: CRC Press.

Winston, W. L. (2004). *Operations research* (4th ed.). Belbont: Brooks/Cole - Thomson Learning.

Zhang, G., Zhang, L., Song, X., Wang, Y., & Zhou, C. (2018). A variable neighborhood search based genetic algorithm for flexible job shop scheduling problem. *Cluster Computing*(22), 11561-11572. doi:https://doi.org/10.1007/s10586-017-1420-4

Appendix I. Determine the product route

Every product has a standard route in the mixing department. A route includes the mixer, order quantity, and mixing steps. The mixing steps state the mixing orders of the ingredients. Table A provides a fictive example of a product route. It shows the optimal order quantity, calculated by the process engineer, the minimum order quantity (MOQ) and the incremental order quantity (IOQ), which depend on agreements with the customer. The IOQ often depends on the bag or pallet size of the finished product.

Table A: Example of a product route used for mixing

Description	**Information**				
Product	Noodle soup mix				
Optimal order quantity	1000kg				
Minimal order quantity	800kg				
Incremental order quantity	25kg				
Ingredients	Salt	Paprika	Pepper	Oil	Noodles
Ingredient steps	1	1	1	2	3

The standard route is composed by a process engineer using an Excel tool and the engineer's knowledge of ingredient properties and mixers. The process engineer designs the route by first meeting the restrictions of the product. For example, adding liquid after mixing most dry ingredients. The process engineer then checks whether the customer demand is satisfied, and a suitable mixer is assigned (a mixer must not be too full or too empty). After developing a feasible route, the process engineer optimizes it. Optimization can include switching ingredients from steps, switching to another size mixer, or changing the quantity made.

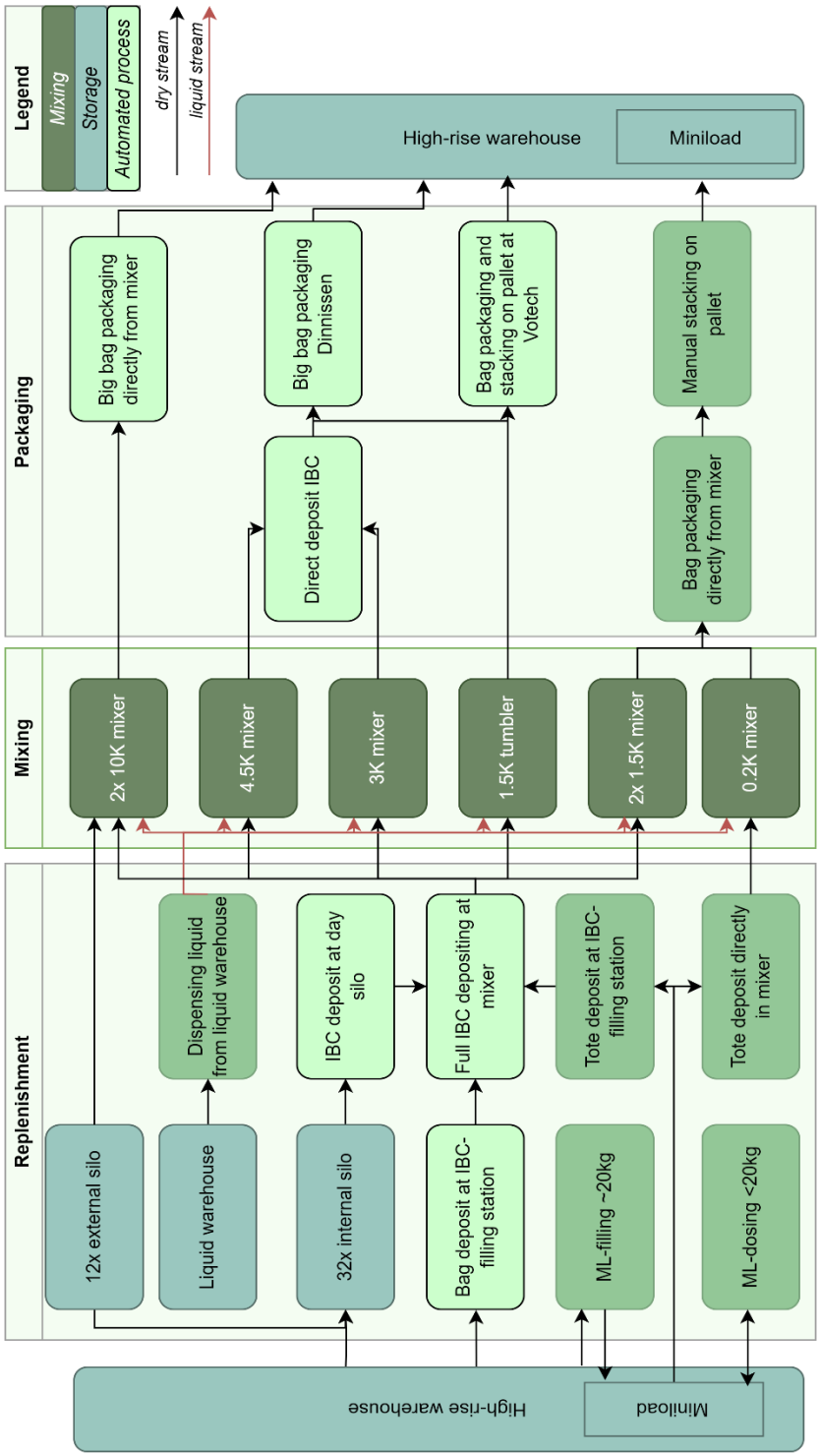
Table B shows an example of switching the ingredient steps. In this table, we see a fictive example of a mix for noodle soup. For almost every mixture, the dry (powder) ingredients are mixed first, the liquid is added and mixed through, and finally, the ingredients that need to be mixed last. In scenario 1 (columns 3 to 5) in the example, the route is designed using this method. The dry powders are added in the first step, liquid in the second, and noodles last to prevent them from breaking in the mixer. This scenario requires 4 IBCs: 3 IBCs in the first step and 1 in the third (for adding liquid IBCs are not used).

Table B: Example of switching ingredients.

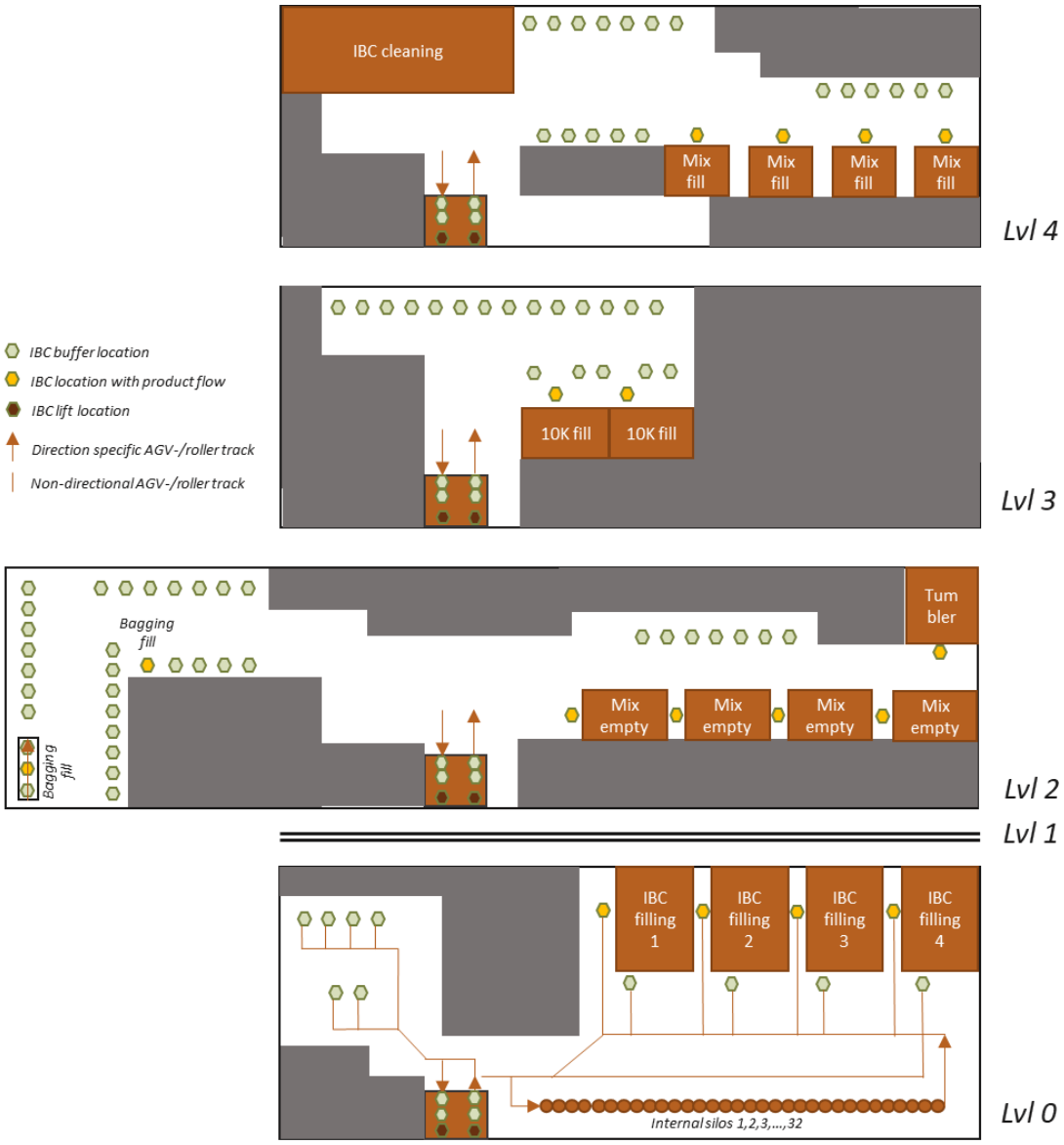
Ingredient		**Scenario 1**			**Scenario 2**		
Name	**IBCs needed**	**Step**	**Nr. IBCs step 1**	**Nr. IBCs step 2**	**Step**	**Nr. IBCs step 1**	**Nr. IBCs step 2**
Salt	0.3	1	0.3		2		0.3
Paprika powder	0.9	1	0.9		1	0.9	
Pepper	0.9	1	0.9		1	0.9	
Oil	-	2	-	-	2	-	-
Noodles (only after liquid)	0.2	3		0.2	3		0.2
Total per step			2.1 → 3	0.2 → 1		1.8 → 2	0.5 → 1
Total per scenario		4			3		

In scenario 2 (columns 6 to 8), the salt switches to process step 3. This relatively small quantity can fit in the remaining space in the IBC used in step 3. This change effectively reduces the required number of IBCs from 4 to 3.

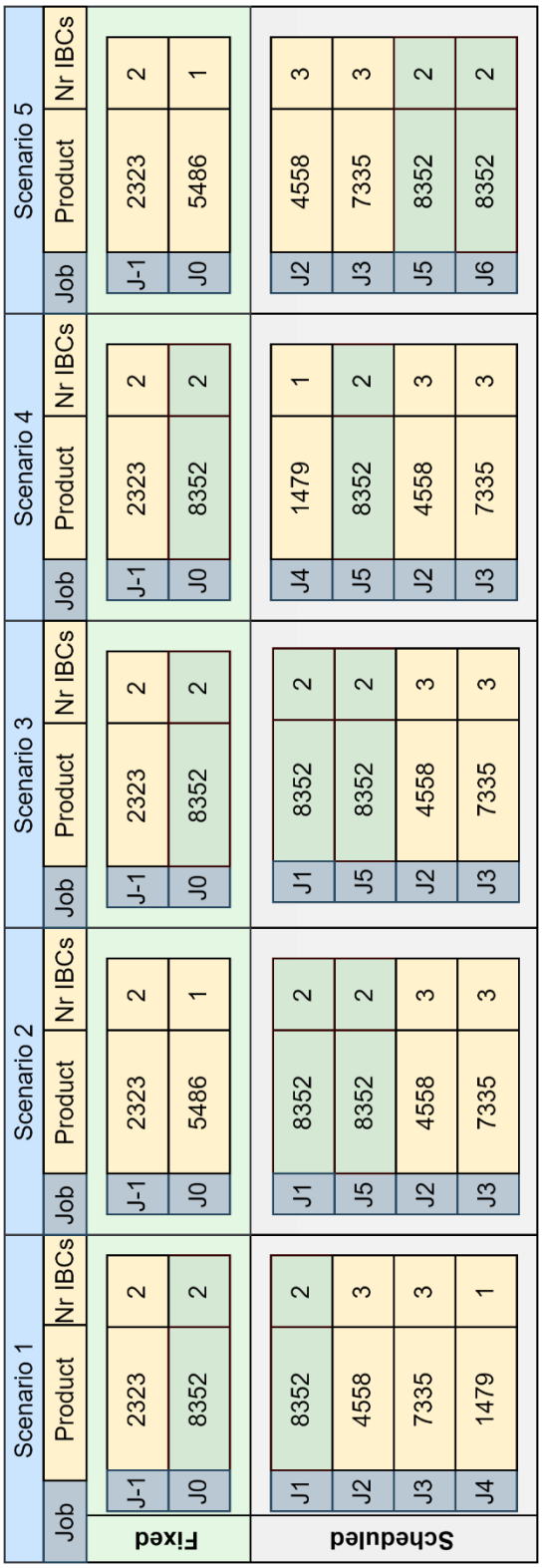
Appendix II. Processes within the mixing department



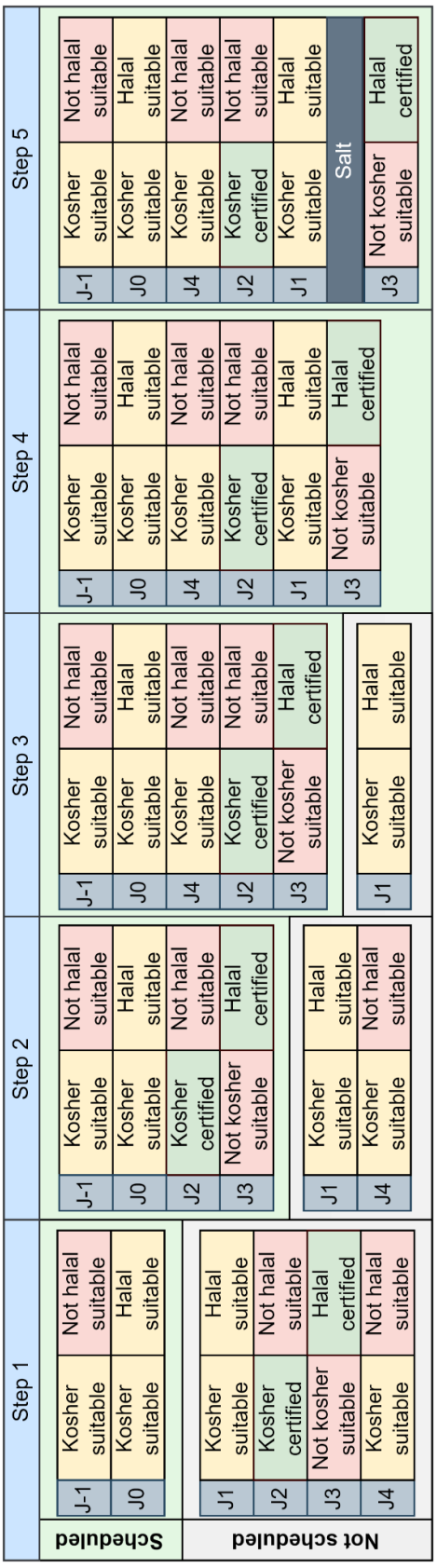
Appendix III. Simplified map of the mixing department



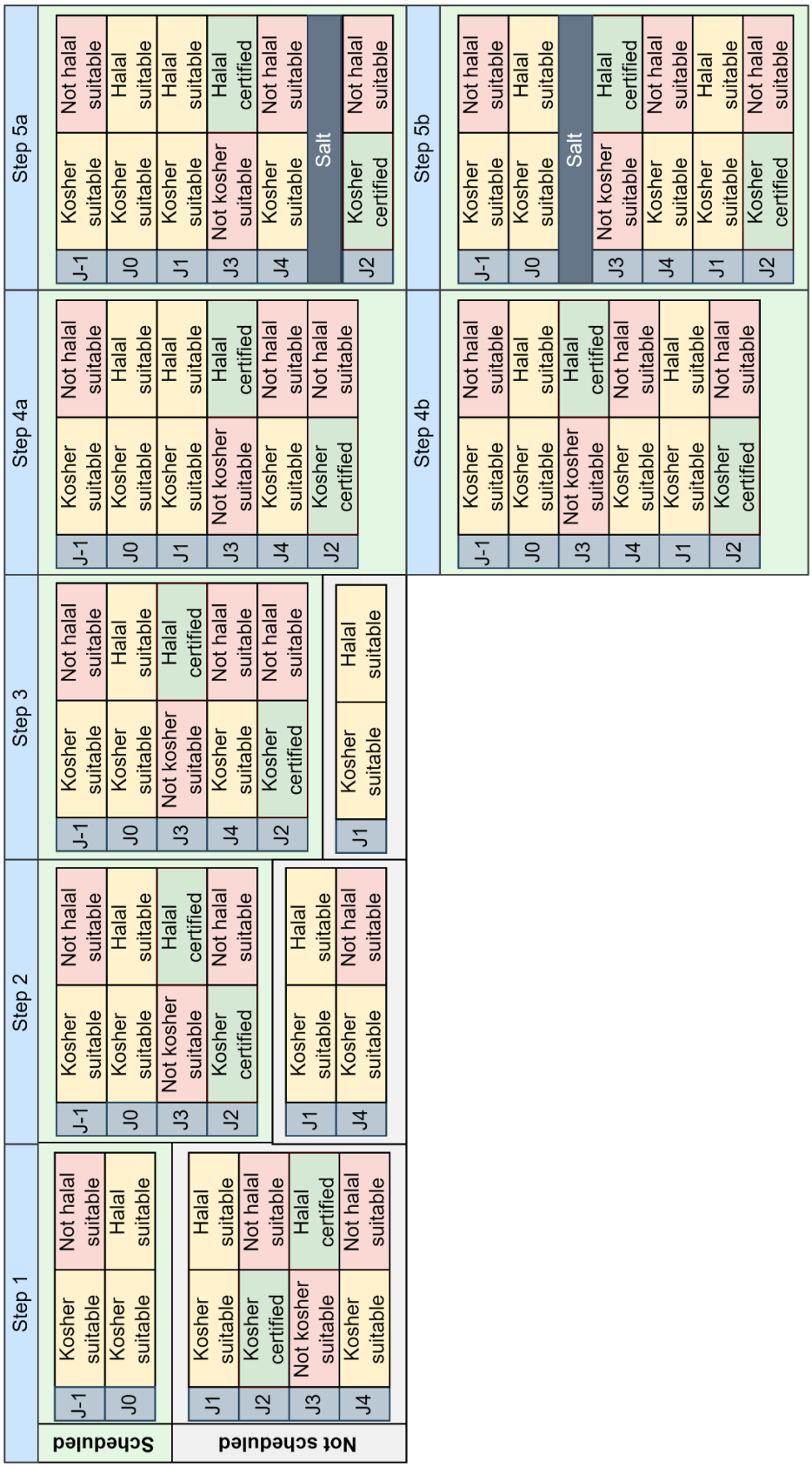
Appendix IV. Examples of schedules to illustrate when a discount for a sequence of the same product



Appendix V. Examples of schedules to illustrate when to give a discount for a sequence of the same product.



Appendix VI. Example calculation of heuristic for minimal dummy jobs, where kosher is planned before halal



Appendix VII. Example of possible schedules when there are halal, kosher, and halal & kosher certified jobs present

Schedule 1

J-1	Kosher suitable	Not halal suitable
J0	Kosher suitable	Not halal suitable
Stop		
J6	Kosher certified	Halal certified
J1	Kosher certified	Not halal suitable
J2	Kosher certified	Halal suitable
J3	Not halal suitable	Halal suitable
J4	Kosher suitable	Halal certified
J5	Kosher suitable	Halal certified

Schedule 2

J-1	Kosher suitable	Not halal suitable
J0	Kosher suitable	Not halal suitable
Stop		
J6	Kosher certified	Halal certified
J3	Not halal suitable	Halal suitable
J4	Kosher suitable	Halal certified
J5	Kosher suitable	Halal certified
J1	Kosher certified	Not halal suitable
J2	Kosher certified	Halal suitable

Schedule 3

J-1	Kosher suitable	Not halal suitable
J0	Kosher suitable	Not halal suitable
J1	Kosher certified	Not halal suitable
J2	Kosher certified	Halal suitable
Salt		
J6	Kosher certified	Halal certified
J3	Not halal suitable	Halal suitable
J4	Kosher suitable	Halal certified
J5	Kosher suitable	Halal certified

Schedule 4

J-1	Kosher suitable	Not halal suitable
J0	Kosher suitable	Not halal suitable
Salt		
J3	Not halal suitable	Halal suitable
J4	Kosher suitable	Halal certified
J5	Kosher suitable	Halal certified
J6	Kosher certified	Halal certified
J1	Kosher certified	Not halal suitable
J2	Kosher certified	Halal suitable

Schedule 5

J-1	Kosher suitable	Not halal suitable
J0	Kosher suitable	Not halal suitable
J1	Kosher certified	Not halal suitable
J2	Kosher certified	Halal suitable
J3	Not halal suitable	Halal suitable
J4	Kosher suitable	Halal certified
J5	Kosher suitable	Halal certified
J6	Kosher certified	Halal certified

Schedule 6

J-1	Kosher suitable	Not halal suitable
J0	Kosher suitable	Not halal suitable
Salt		
J3	Not halal suitable	Halal suitable
J4	Kosher suitable	Halal certified
J5	Kosher suitable	Halal certified
J1	Kosher certified	Not halal suitable
J2	Kosher certified	Halal suitable
J6	Kosher certified	Halal certified

Appendix VIII. Contamination matrix

A value of 1 in a cell refers to a necessary cleaning, and a 0 to no cleaning is necessary.

From \ To	white	white / green	grey	green	yellow	dark yellow	orange	red	light brown	brown	dark brown	black
white	0	0	0	0	0	0	0	0	0	0	0	0
white / green	0	0	0	0	0	0	0	0	0	0	0	0
grey	0	0	0	0	0	0	0	0	0	0	0	0
green	1	1	1	0	0	0	0	0	0	0	0	0
yellow	1	1	1	0	0	0	0	0	0	0	0	0
dark yellow	1	1	1	0	0	0	0	0	0	0	0	0
orange	1	1	1	1	1	1	0	0	1	0	0	0
red	1	1	1	1	1	1	0	0	1	1	1	1
light brown	1	1	1	0	0	0	0	0	0	0	0	0
brown	1	1	1	1	1	1	0	0	0	0	0	0
dark brown	1	1	1	1	1	1	1	0	0	0	0	0
black	1	1	1	1	1	1	1	1	1	0	0	0

Appendix IX. Enhancing the processing and set-up time

This Appendix provides the method for enhancing the norms to create a more realistic situation. The times used are processing time and set-up time. The set-up time exists out of cleaning time, time for changing a colour or product, and dummy jobs.

The adjustment increases all the times with a factor expressed as a percentage:

$$Adjusted\ Time = Original\ Time * \frac{100\% + Factor}{100\%}$$

We experiment with enhancing this factor until we have an IBC level comparable to the situation at Euroma. For this experiment, we use NNH and steepest descent. This does not require any adjustment of parameters in the heuristics. As weights, we use 50%, 20%, 15%, and 10% for makespan, tardiness, discount for sequences, and penalty for the dummy jobs, respectively.

We experiment over 1 period of 20 days with different factors. The factor increases with steps of 5%. Figure A displays the result of four of the simulations. It shows the average number of IBCs per day per simulation and the average number of IBCs in the buffer at Euroma.

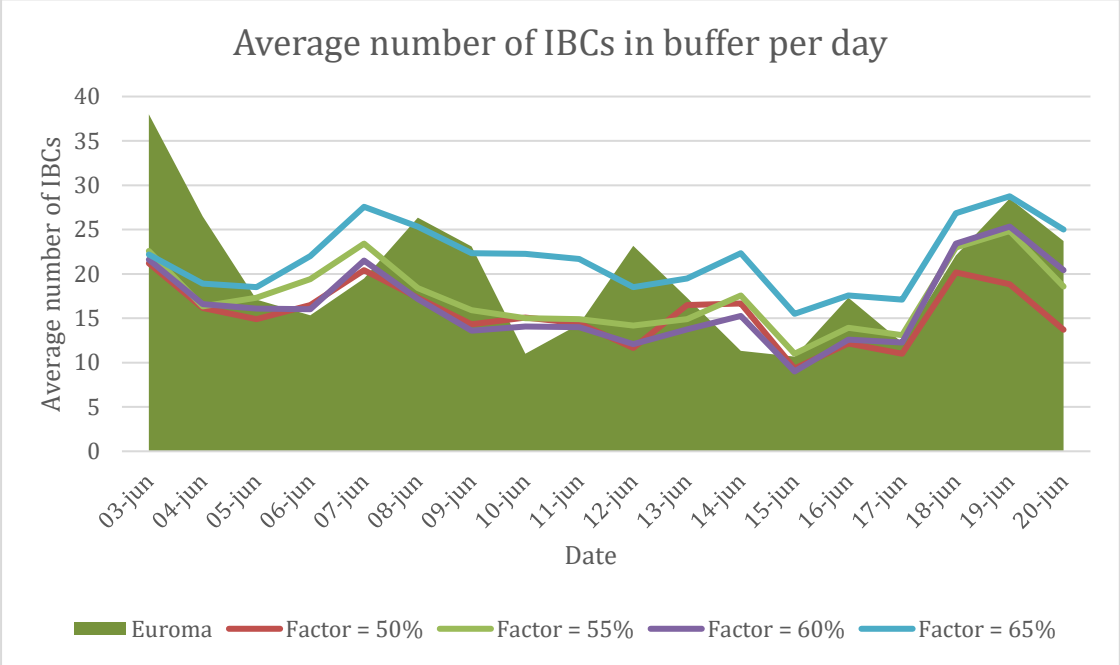


Figure A: Comparing the number of IBCs over time with different factors to increase the norm times.

To easily compare the results, the averages from the simulations subtract from the number of IBCs waiting at Euroma at that time. Table C shows the deviation of the four factors with the sum closest to zero and the absolute deviation. In both instances, the factor of 60% gives the lowest deviation from the situation at Euroma. This indicates that this simulation is most representative of the situation at Euroma. Therefore, we select this parameter, $Factor = 60\%$, for the experiments.

Date	Average number of IBCs waiting per factor				The absolute average number of IBCs waiting per factor			
Experiment	50%	55%	60%	65%	50%	55%	60%	65%
03-jun	16.8	15.4	16.4	15.8	16.8	15.4	16.4	15.8
04-jun	10.3	10.2	9.9	7.6	10.3	10.2	9.9	7.6
05-jun	2.2	-0.3	1.0	-1.4	2.2	0.3	1.0	1.4
06-jun	-1.2	-4.1	-0.7	-6.7	1.2	4.1	0.7	6.7
07-jun	-0.9	-3.9	-2.0	-8.1	0.9	3.9	2.0	8.1
08-jun	9.0	7.9	9.2	1.0	9.0	7.9	9.2	1.0
09-jun	8.8	7.2	9.5	0.8	8.8	7.2	9.5	0.8
10-jun	-4.1	-4.0	-3.1	-11.3	4.1	4.0	3.1	11.3
11-jun	-0.1	-0.6	0.3	-7.3	0.1	0.6	0.3	7.3
12-jun	11.5	9.0	11.1	4.7	11.5	9.0	11.1	4.7
13-jun	0.8	2.4	3.6	-2.2	0.8	2.4	3.6	2.2
14-jun	-5.3	-6.3	-3.9	-11.0	5.3	6.3	3.9	11.0
15-jun	1.3	-0.3	1.7	-4.8	1.3	0.3	1.7	4.8
16-jun	5.2	3.4	4.8	-0.3	5.2	3.4	4.8	0.3
17-jun	1.3	-0.8	0.0	-4.8	1.3	0.8	0.0	4.8
18-jun	1.9	-0.9	-1.3	-4.8	1.9	0.9	1.3	4.8
19-jun	9.7	3.7	3.2	-0.3	9.7	3.7	3.2	0.3
20-jun	10.0	5.1	3.3	-1.3	10.0	5.1	3.3	1.3

Table C: Impact on enhancing demand and IBCs in the buffer.

Appendix X. Results weight selection objective

Weight				Votech					
$W_{makespan}$	W_{tardy}	W_{dummy}	W_{seq}	$F_{makespan}$	F_{tardy}	F_{dummy}	F_{seq}	IBCs	Dummy time (h)
0.4	0.1	0.15	0.35	1.49	7.16	0.007	0.000	16.3	0.051
0.4	0.125	0.125	0.35	1.50	7.14	0.007	0.000	16.2	0.093
0.4	0.125	0.15	0.325	1.50	7.14	0.007	0.000	16.2	0.093
0.425	0.075	0.15	0.35	1.47	7.84	0.004	0.000	15.6	0.000
0.425	0.1	0.125	0.35	1.37	7.63	0.042	0.000	16.0	0.019
0.425	0.1	0.15	0.325	1.37	7.63	0.042	0.000	16.0	0.019
0.425	0.125	0.1	0.35	1.50	7.13	0.007	0.000	16.4	0.090
0.425	0.125	0.125	0.325	1.50	7.13	0.007	0.000	16.4	0.090
0.425	0.125	0.15	0.3	1.50	7.14	0.007	0.000	16.2	0.093
0.45	0.05	0.15	0.35	1.41	8.28	0.007	0.000	16.3	0.071
0.45	0.075	0.125	0.35	1.48	9.07	0.065	0.000	17.4	0.002
0.45	0.075	0.15	0.325	1.48	9.09	0.065	0.000	17.4	0.002
0.45	0.1	0.1	0.35	1.37	7.62	0.042	0.000	16.0	0.023
0.45	0.1	0.125	0.325	1.37	7.62	0.042	0.000	16.0	0.023
0.45	0.1	0.15	0.3	1.37	7.62	0.042	0.000	16.0	0.023
0.45	0.125	0.075	0.35	1.51	7.63	0.055	0.001	16.5	0.096
0.45	0.125	0.1	0.325	1.51	7.63	0.055	0.001	16.5	0.096
0.45	0.125	0.125	0.3	1.51	7.63	0.055	0.001	16.5	0.096
0.45	0.125	0.15	0.275	1.51	7.63	0.055	0.001	16.5	0.096
0.475	0.025	0.15	0.35	1.46	9.15	0.004	0.000	16.5	0.000
0.475	0.05	0.125	0.35	1.41	8.29	0.007	0.000	16.3	0.071
0.475	0.05	0.15	0.325	1.41	8.29	0.007	0.000	16.3	0.071
0.475	0.075	0.1	0.35	1.37	7.46	0.004	0.002	15.5	0.022
0.475	0.075	0.125	0.325	1.37	7.46	0.004	0.002	15.5	0.022
0.475	0.075	0.15	0.3	1.37	7.46	0.004	0.002	15.5	0.022
0.475	0.1	0.075	0.35	1.43	8.53	0.042	0.000	16.7	0.000
0.475	0.1	0.1	0.325	1.43	8.53	0.042	0.000	16.7	0.000
0.475	0.1	0.125	0.3	1.43	8.53	0.042	0.000	16.7	0.000
0.475	0.1	0.15	0.275	1.43	8.53	0.042	0.000	16.7	0.000
0.475	0.125	0.05	0.35	1.53	8.12	0.058	0.001	16.8	0.100
0.475	0.125	0.075	0.325	1.53	8.12	0.058	0.001	16.8	0.100
0.475	0.125	0.1	0.3	1.53	8.12	0.058	0.001	16.8	0.100
0.475	0.125	0.125	0.275	1.53	8.13	0.058	0.001	16.8	0.088
0.475	0.125	0.15	0.25	1.55	8.17	0.058	0.002	16.9	0.093
0.5	0.025	0.125	0.35	1.45	9.03	0.004	0.000	16.3	0.000
0.5	0.025	0.15	0.325	1.45	9.03	0.004	0.000	16.3	0.000
0.5	0.05	0.1	0.35	1.48	9.48	0.004	0.000	16.9	0.000
0.5	0.05	0.125	0.325	1.48	9.48	0.004	0.000	16.9	0.000
0.5	0.05	0.15	0.3	1.48	9.48	0.004	0.000	16.9	0.000
0.5	0.075	0.075	0.35	1.36	7.16	0.004	0.001	15.5	0.000
0.5	0.075	0.1	0.325	1.36	7.16	0.004	0.001	15.5	0.000
0.5	0.075	0.125	0.3	1.36	7.16	0.004	0.001	15.5	0.000
0.5	0.075	0.15	0.275	1.36	7.16	0.004	0.001	15.5	0.000
0.5	0.1	0.05	0.35	1.38	7.77	0.042	0.000	15.8	0.018
0.5	0.1	0.075	0.325	1.38	7.77	0.042	0.000	15.8	0.018
0.5	0.1	0.1	0.3	1.38	7.77	0.042	0.000	15.8	0.018

0.5	0.1	0.125	0.275	1.38	7.77	0.042	0.000	15.8	0.018
0.5	0.1	0.15	0.25	1.38	7.77	0.042	0.000	15.8	0.018
0.5	0.125	0.05	0.325	1.49	7.16	0.007	0.000	16.3	0.052
0.5	0.125	0.075	0.3	1.49	7.16	0.007	0.000	16.3	0.052
0.5	0.125	0.1	0.275	1.49	7.16	0.007	0.000	16.3	0.051
0.5	0.125	0.125	0.25	1.49	7.16	0.007	0.000	16.3	0.051
0.525	0.025	0.1	0.35	1.49	9.81	0.006	0.000	16.9	0.000
0.525	0.025	0.125	0.325	1.49	9.81	0.006	0.000	16.9	0.000
0.525	0.025	0.15	0.3	1.49	9.81	0.006	0.000	16.9	0.000
0.525	0.05	0.075	0.35	1.48	9.47	0.004	0.000	16.6	0.048
0.525	0.05	0.1	0.325	1.48	9.47	0.004	0.000	16.6	0.048
0.525	0.05	0.125	0.3	1.48	9.47	0.004	0.000	16.6	0.048
0.525	0.05	0.15	0.275	1.48	9.48	0.004	0.000	16.6	0.048
0.525	0.075	0.05	0.35	1.36	7.16	0.004	0.001	15.5	0.073
0.525	0.075	0.075	0.325	1.36	7.16	0.004	0.001	15.5	0.073
0.525	0.075	0.1	0.3	1.36	7.16	0.004	0.001	15.5	0.073
0.525	0.075	0.125	0.275	1.36	7.16	0.004	0.001	15.5	0.073
0.525	0.075	0.15	0.25	1.36	7.16	0.004	0.001	15.5	0.073
0.525	0.1	0.05	0.325	1.37	7.90	0.042	0.000	16.2	0.002
0.525	0.1	0.075	0.3	1.37	7.90	0.042	0.000	16.2	0.002
0.525	0.1	0.1	0.275	1.37	7.90	0.042	0.000	16.2	0.002
0.525	0.1	0.125	0.25	1.37	7.90	0.042	0.000	16.2	0.002
0.525	0.125	0.05	0.3	1.38	7.63	0.042	0.000	16.2	0.014
0.525	0.125	0.075	0.275	1.38	7.63	0.042	0.000	16.2	0.014
0.525	0.125	0.1	0.25	1.38	7.63	0.042	0.000	16.2	0.014
0.55	0.025	0.075	0.35	1.44	8.86	0.004	0.003	16.3	0.075
0.55	0.025	0.1	0.325	1.44	8.87	0.004	0.002	16.3	0.064
0.55	0.025	0.125	0.3	1.44	8.87	0.004	0.001	16.3	0.060
0.55	0.025	0.15	0.275	1.48	10.27	0.004	0.001	16.8	0.073
0.55	0.05	0.05	0.35	1.46	9.13	0.004	0.000	16.6	0.000
0.55	0.05	0.075	0.325	1.46	9.13	0.004	0.000	16.6	0.000
0.55	0.05	0.1	0.3	1.46	9.13	0.004	0.000	16.6	0.000
0.55	0.05	0.125	0.275	1.46	9.13	0.004	0.000	16.6	0.000
0.55	0.05	0.15	0.25	1.46	9.13	0.004	0.000	16.6	0.000
0.55	0.075	0.05	0.325	1.39	7.76	0.007	0.000	16.0	0.063
0.55	0.075	0.075	0.3	1.39	7.76	0.007	0.000	16.0	0.063
0.55	0.075	0.1	0.275	1.39	7.76	0.007	0.000	16.0	0.063
0.55	0.075	0.125	0.25	1.39	7.76	0.007	0.000	16.0	0.063
0.55	0.1	0.05	0.3	1.44	7.66	0.004	0.000	15.3	0.000
0.55	0.1	0.075	0.275	1.44	7.66	0.004	0.000	15.3	0.000
0.55	0.1	0.1	0.25	1.44	7.66	0.004	0.000	15.3	0.000
0.55	0.125	0.05	0.275	1.37	7.63	0.042	0.000	16.0	0.029
0.55	0.125	0.075	0.25	1.37	7.63	0.042	0.000	16.0	0.029

Appendix XI. Increasing demand

This Appendix provides the method for generating input for the experiments when a higher demand is required. The incoming jobs correlate, as they come from mixers that must follow the same restrictions as the packaging machines. An exception is the Tumbler, where the product stays in the IBC and does not go into a mixer. As this is only one of four (and the smaller one) mixers, we generalize the statement of the correlation.

To generate more demand in the same period we can create more demand in the same period or speed up the period with the same demand. We choose the second option to not lose the correlation between the jobs. The following steps show the generation of increased demand. Table D shows an example of two jobs.

1. Choose a base time. This is the starting point in time that does not change. We select this to be the starting time of the simulation.
2. Create a delta for every time recorded for a job.
3. Reduce (or increase if we want to decrease demand) the delta with the desired factor. In the example, we increase demand by 20%. The new delta is, therefore, generated by $(100\% - 80\%) * original\ delta$.
4. Tstart time value increases by the new delta.

Job	In Mixer	In Buffer	In Machine	Out Machine
Original values				
J00058219	2023-06-02 01:07	2023-06-02 05:31	2023-06-04 08:18	2023-06-04 09:10
J00056134	2023-06-02 05:43	2023-06-02 08:07	2023-06-03 21:53	2023-06-03 23:07
Delta (in days): Start Time = 2023-06-01 12:00				
J00058219	0.55	0.73	2.85	2.88
J00056134	0.74	0.84	2.41	2.46
New Delta: Factor of -20%				
J00058219	0.44	0.58	2.28	2.31
J00056134	0.59	0.67	1.93	1.97
New values				
J00058219	2023-06-01 22:30	2023-06-02 02:01	2023-06-03 18:38	2023-06-03 19:20
J00056134	2023-06-02 02:10	2023-06-02 04:06	2023-06-03 10:19	2023-06-03 11:17

Table D: Generation of increased demand.

Appendix XII. Example complete schedule

Machine	Job	Article	Article Description	Time in Mixer	Time in buffer	Status	Planned time start	Planned time stop	NrIBCs	Color	Halal certification	Kosher certification	Allergens
Z410	J00058025	59883	Jus Rundvlees	9-06-23 17:51	9-06-23 20:09	Stopped	9-06-23 23:50	10-06-23 1:14	2	Bruin	Halal suitable	Trefa	
Z410	J00058189	51108	Oosterse melange	9-06-23 1:41	9-06-23 3:18	Active	10-06-23 1:38	10-06-23 2:55	2	Lichtbruin	Halal suitable	Kosher suitable	
Z410	J00057271	65893	Oosterse mix	9-06-23 3:38	9-06-23 5:26	Released	10-06-23 3:11	10-06-23 4:43	2	Lichtbruin	Halal suitable	Kosher suitable	
Z410	J00058024	59417	BBQ classic	9-06-23 16:49	9-06-23 18:56	Released	10-06-23 5:07	10-06-23 6:51	2	Oranje	Halal	Kosher	
Z410	J00058449	60931	Fajita mix	10-06-23 1:10		Mixer	10-06-23 7:15	10-06-23 8:51	2	Rood	Halal suitable	Kosher suitable	
Z410	J00058432	52426	Bolognese	9-06-23 6:15	9-06-23 7:57	Released	10-06-23 9:15	10-06-23 11:11	3	Oranje	Halal suitable	Kosher suitable	So;So
Z410	S00000001	67106	Dry Cleaning				10-06-23 11:11	10-06-23 12:47					
Z410	J00058415	65139	Curry Japans	9-06-23 16:35	9-06-23 17:31	Released	10-06-23 12:47	10-06-23 14:28	2	Groen	Haram	Trefa	Gl;Me;So;So
Z410	J00058250	55158	Curry Geel	10-06-23 0:57		Mixer	10-06-23 14:52	10-06-23 16:01	2	Donkergeel	Halal suitable	Kosher suitable	Gl;Me;So;So
Z420	J00058420	66204	Champignons saus	9-06-23 16:25	9-06-23 18:04	Stopped	9-06-23 23:57	10-06-23 0:58	1	Geel	Halal suitable	Trefa	Mo
Z420	J00058418	66204	Champignons saus	9-06-23 20:33	9-06-23 23:02	Active	10-06-23 0:58	10-06-23 1:59	1	Geel	Halal suitable	Trefa	Mo
Z420	J00058245	65745	Nacho mix	9-06-23 19:54	9-06-23 20:47	Released	10-06-23 6:04	10-06-23 6:43	1	Geel	Halal suitable	Kosher suitable	Mo
Z420	J00058244	65745	Nacho mix	9-06-23 16:31	9-06-23 18:52	Released	10-06-23 6:43	10-06-23 7:22	1	Geel	Halal suitable	Kosher suitable	Mo
Z420	S00000002	67107	Wet Cleaning				10-06-23 7:22	10-06-23 10:34					
Z420	J00058525	61775	Nasi kruiden	9-06-23 18:06	9-06-23 19:42	Released	10-06-23 10:34	10-06-23 11:16	1	Geel	Halal	Kosher suitable	So
Z420	J00058526	61775	Nasi kruiden	10-06-23 1:03	10-06-23 2:45	Mixer	10-06-23 11:16	10-06-23 11:58	1	Geel	Halal	Kosher suitable	So