Reducing the Work-In-Process inventory in a cleanroom production environment

Bart Oerbekke

Industrial Engineering and Management

UNIVERSITY OF TWENTE.

STRENGTH THROUGH COOPERATION

University of Twente

Faculty of Behavioural, Management and Social Sciences
Industrial Engineering and Management

Reducing the Work-In-Process inventory in a cleanroom production environment

Author B.G. Oerbekke

Company supervisor G. Mueller-Ruhlandt
VDL Enabling Technologies Group

Leading supervisor Dr. E. Topan
University of Twente

Secondary Supervisor Dr. I. Seyran Topan
University of Twente

2-5 2025

Management Summary

This research is conducted at VDL ETG Almelo (VLD ETGA), which is part of the VDL groep and operates in the semiconductor industry. This research is executed within the Systems-2 division of VDL ETGA. This division manufactures four different modules for a producer of photolithographic equipment in a cleanroom environment compliant with ISO 6 norms. The Systems-2 division has been in existence since 2019. After two years of developing the cleanroom environment, cleaning processes and manufacturing processes Systems-2 delivered its first modules to their customer in 2021. This manufacturing rate of 4 modules per year was also the forecast for 2022 but due to an increase in demand for the modules, the manufacturing rate increased to an average of 10 modules per year, or 0.8 modules per month. It is expected that in the upcoming years the demand for modules will further increase, resulting in a manufacturing rate up to 20 modules per year, or 1,67 modules per month. The cleanrooms are originally designed for manufacturing the modules without taking inventory space into account. So, with the target in mind to obtain a manufacturing rate, more materials must be delivered to the cleanroom meaning that it is now crucial to utilize the cleanroom space for its initial purpose which is assembling modules. The objective of this research is to reduce the Work-In-Process inventory while also improving the flow of materials. The research question reads as follows:

How can the WIP inventory at the production, cleaning and unpacking area be controlled while taking maximum capacity at the cleaning and unpacking area, and different types of load carriers into account?

As a solution approach we developed a mixed-integer-programming model based on the Multi-Level Capacitated Lot Sizing Problem to balance the tradeoff between delivery moments and inventory levels. By incorporating demand scenarios we have modelled the uncertainty in demand. The delivery of items to the production area is modelled as a setup with a maximum available capacity, where a distinction is made between the cleaning and unpacking area based on the load carrier of an item. The model is able to decide when an item must be delivered to the production area by minimizing the setup cost, inventory cost and potential overtime cost if the maximum capacity is exceeded.

The model is unable to find an optimal solution in polynomial time, therefore a solution approach has been developed based on a Fix-and-Optimize approach. In our developed approach we destroy a part of the solution based on the interrelation between setup variables. The destroyed part of the solution creates a subproblem that can be optimized by inserting the partial solution into the MIP. By experimenting with levels of interrelatedness between variables, we managed to find a configuration for the solution approach that consistently solves the subproblems to optimality in 4,75 seconds on average while finding improvements to the solution. We compare our algorithm with a simplified algorithm and show that the decisions made during the development of our algorithm result in better solutions that are obtained in a short time.

We have applied the model to two modules and found a new setup policies for each module. Both new setup policies make use of additional delivery moments resulting in lower inventory cost and improvements in the capacity usage. To get an idea how a new setup policy compares to the current situation, we have inserted the current setup policy in the MIP model. Figure 1 shows that the new policy results in lower inventory cost in every period when materials arrive either early, late or at their planned time.

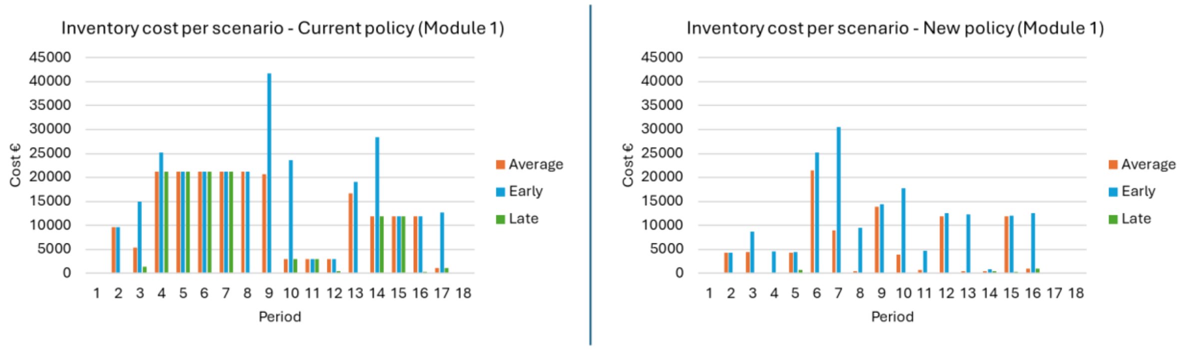


Figure 1: Inventory cost per scenario of the current policy and new policy for Module 1

In addition to the lower inventory cost, the additional delivery moments also result in improvements in the capacity usage. Figure 2 shows that the capacity usage in the new policy does not exceed the maximum capacity. The capacity usage is also more spread over the periods, resulting in less variation in workload at both the unpacking and cleaning area.

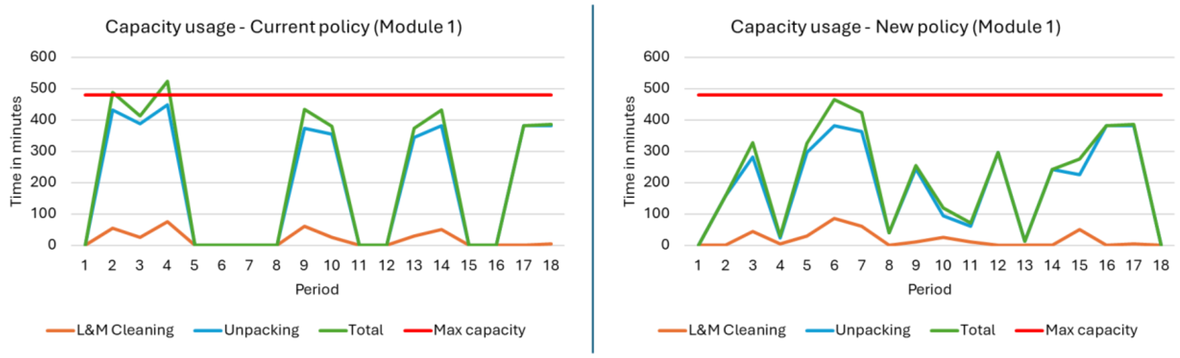


Figure 2: Capacity usage of the current policy and new policy for module 1

Systems-2 must now apply this model to the remaining modules that are not included in this research to get a setup policy for every module it produces. Before the new setup policies can be implemented, the logistics department must first cooperate with the Factory Engineers to create new material sets based on the new setup policies. The Factory Engineers must be consulted to correctly allocate the materials in the material sets. Once the new material sets are designed, the new setup policies can be used. To monitor and continuously improve the material and production planning process, Systems-2 must create an alignment between the status of the production process and their ERP-system.

Preface

Dear reader, I am pleased to present my master's thesis, '*Reducing the Work-In-Process inventory in a cleanroom production environment',* which marks the final milestone of my studies in the Master's program Industrial Engineering and Management with a specialization in Production, Logistics, and management at the University of Twente.

I am grateful for the opportunity and assignment I received at VDL ETG Almelo. I reflect on my time at VDL ETG with great appreciation. I would like to express my gratitude to Gils Mueller-Ruhlandt, my company supervisor, for his guidance and support. I also want to thank Engin Topan for his invaluable guidance, insightful feedback and sincere interest as my first supervisor. I want to thank Ipek Seyran Topan as well for the feedback and recommendations as my second supervisor.

I would like to thank my family for their continuous support throughout my time as a student. Their support allowed me to maintain a healthy balance between my studies, social life, and personal interests. Finally, I am deeply grateful to my girlfriend for always being there for me and supporting me throughout this period.

Thank you for taking the time to read my thesis. I hope it offers valuable insights and I wish you a pleasant reading experience.

Bart Oerbekke

Enschede, May 2025

Contents

List of Figures

List of Tables

Acronyms

WIP	Work-in-Process
TT	Throughput Time
XLOF	eXtra Large Outgassing Facility
PCF	Precision Cleaning Facility
KPI	Key Performance Indicator
PA	Production Assistant
MAS	Montage Afloop Schema – Assembly Completion Schedule
FTE	Full-Time Equivalent
MLCLSP	Multi-Level Capacitated Lot-Sizing Problem
FO	Fix and Optimize
SAA	Sample Average Approximation
VNS	Variable Neighborhood Search
WACC	Weighted Average Cost of Capital
MIP	Mixed-Integer-Programming
LP	Linear Programming

1 Introduction

In this chapter we introduce the research design. In Section 1.1 the company is introduced followed by a more in depth description of the manufacturing process at the Systems-2 division, where this research takes place. In Sections 1.2, 1.3 and 1.4 the research motivation, problem description and problem approach are presented. In Section 1.5 the research questions are described.

1.1 Company description

Van Der Leegte Groep (VDL Groep) is a family business that has been in existence since 1953. It started as a metal industry and construction site and the company's first clients were Philips and DAF Trucks. The initial work of the first five employees involved turning, milling, and drilling, as well as punching, welding, and soldering in series production. Currently, VDL Groep has grown into an international company that consists of more than one hundred companies with over 14.000 employees in 20 countries. The VDL Groep operates in five areas: High tech, Mobility, Energy, Infratech, and Foodtech and has a combined turn-over of €6.354 billion (VDL Groep, 2024).

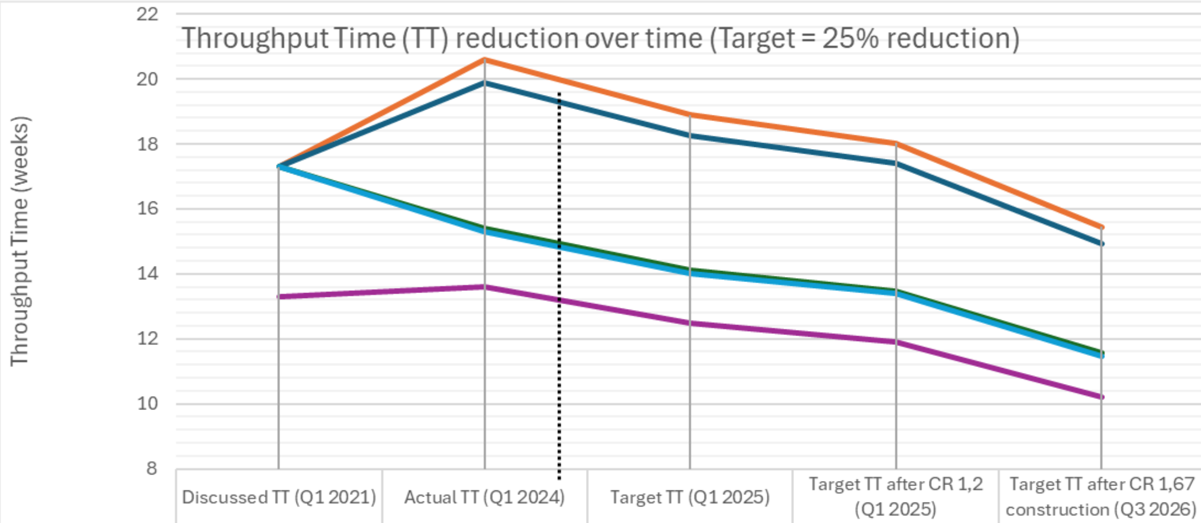
VDL Enabling Technologies Group is part of the VDL Groep and operates in the high-tech area and is founded in 1900 as Philips Machinefabriek. VDL ETG is since 2006 part of the VDL Groep. As a global development and manufacturing company it collaborates with leading companies in the field of high-tech systems and modules with locations in Singapore, China, Switzerland, The United States of America, and The Netherlands. VDL ETG operates in four businesses: (i) the analytical industry for which high value subsystems and analytical equipment are designed and manufactured to improve the pace and quality of analyses, (ii) the medical industry, where it aims for more efficient diagnostics and treatment by supporting clients in every product life cycle, (iii) science and industry to continuously develop and improve in the field of physics, chemistry, optics, mechatronics, and data analysis, and at last, (iv) the semiconductor industry, better known as the chip industry, where VDL ETG has been collaborating intensively for decades to design and manufacture modules and systems for a producer of photolithographic equipment.

The semiconductor industry is also the industry where VDL ETG Almelo (VDL ETGA) operates. It realizes system integrations of mechatronic (sub)systems and modules for original equipment manufacturers (OEMs) of high-tech capital goods. As a system supplier, their value chain encompasses everything from (co-) design to parts production, assembly, and final qualification. They do this with over 1500 employees, spread over five locations in Almelo and a total of 35.000 m^2 manufacturing space. VDL ETGA can be divided into four divisions: Systems-1, Systems-2, Projects, and Parts.

This research takes place within the Systems-2 division of VDL ETGA. This division manufactures four different modules for a producer of photolithographic equipment. Within Systems-2 every module has its own department accompanied by a Purchasing, Logistics and Quality department. These four modules are manufactured at dedicated workplaces. In the semiconductor industry cleanliness is an important aspect, even the smallest particles can cause defects in the final product. Therefore, these four modules are manufactured in a cleanroom environment compliant with ISO 6 norms. To comply with this ISO 6 norm, a maximum of one million particles of 0,1 µm or 293 particles of 5 µm per cubic meter are allowed.

1.2 Research motivation

The Systems-2 division has been in existence since 2019. After two years of developing the cleanroom environment, cleaning process and manufacturing process it delivered its first modules to their customer in 2021. This manufacturing rate of four modules per year was also the forecast for 2022 but due to an increased demand for the modules it has increased over time to an average of 10 modules per year equal to 0.8 modules per month (move-rate). Figure 3 displays the target throughput time reduction over time which is set at a 25% reduction by quartile 3 in 2026. The goal for 2025 is to assembly modules according to a move-rate of 1.2 and 1.67 in 2026.



	Discussed TT (Q1 2021)	Actual TT (Q1 2024)	Target TT (Q1 2025)	Target TT after CR 1,2 (Q1 2025)	Target TT after CR 1,67 construction (Q3 2026)
Module 1	17,3	20,6	18,9	18,0	15,5
Module 2	17,3	19,9	18,2	17,4	14,9
Module 3A	17,3	15,4	14,1	13,5	11,6
Module 3B	17,3	15,3	14,0	13,4	11,5
Module 4	13,3	13,6	12,5	11,9	10,2

Figure 3: Forecast of Throughput Time (TT) reduction over time

The cleanrooms are originally designed for manufacturing the modules without taking inventory space into account. So, with the target in mind to obtain a higher move rate, more materials must be delivered to the cleanroom meaning that it is now crucial to utilize the cleanroom space for its initial purpose which is assembling modules.

1.3 Problem description

The demand for the modules is expected to grow resulting in an increase in the move rate to 1.2 modules per month in 2025 and 1.67 in 2026. To let that happen, two problems must be addressed and require action. First, extra burnout ovens need to be installed. There are currently two XLOF's (eXtra Large Outgassing Facility) and one PCF (Precision Cleaning Facility) in the cleanroom. This is sufficient for now, but internal calculations show that for a move rate of 1, a minimum three XLOF's and two PCFs required. However, this problem is already addressed by Systems-2 by expanding the current cleanroom such that an extra XLOF and PCF can be installed. Secondly, more cleanroom space is required to manufacture modules simultaneously and move to a series production. A solution for this is to expand the current cleanroom, this however would result in a lot of investment costs. Therefore, Systems-2 investigated other ways to create more production space. It came to the conclusion that there is too much work in process (WIP) inventory at the production, cleaning and unpacking area. Therefore Systems-2 wants to control the WIP inventory at the production area to create more room for the production of their modules and a better flow of materials to the production area. This problem of having too much WIP inventory can be defined as an action problem. An action problem is a discrepancy between the norm and the reality, as perceived by the problem owner (Heerkens & van Winden, 2021). The norm is a lower WIP inventory in the production, cleaning and unpacking area and the reality is that this WIP inventory is too high, as perceived by the Systems-2 division of VDL ETGA. An action problem never occurs on its own. Therefore, a problem cluster has been developed. A problem cluster is used to map all problems with along with their connections. It serves to bring order to the problem context and to identify the core problem (Heerkens & van Winden, 2021). The problem cluster is displayed in Figure 4.

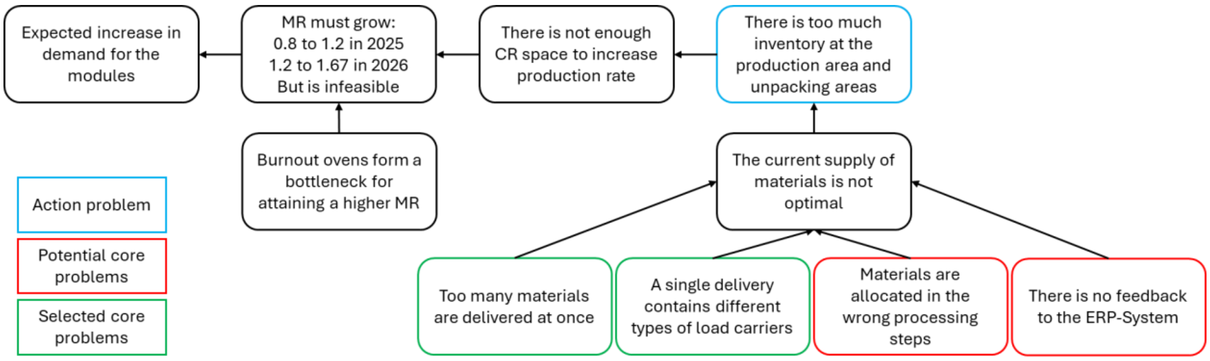


Figure 4: Problem cluster

In the problem cluster it can be seen that there are four core problems related to the action problem:

1. Too many parts are delivered at once
2. A single delivery contains multiple load carriers
3. Parts are allocated in wrong sets
4. There is no alignment between reality and the planned schedule through an ERP-system

Too many materials are delivered at once

When Systems-2 started production, it was important to keep the production running at a continuous rate. Therefore, a lot of materials are assigned to single processing steps and being supplied to the cleanroom ensuring that there is a high material availability at all times. These large sets of materials cause too much WIP inventory inside the cleanroom but in the first few years this has not been a problem due to relatively low production numbers. However, due to the growing move-rate, Systems-2 has come to notice that this WIP inventory at the production, cleaning and unpacking area has become a problem. Having too many materials assigned to a single processing step causes problems for the delivery of materials to the production area. Due to the fact that the production area is in a cleanroom environment, all materials require an unpacking or cleaning process that requires a certain amount of time, and these processes have a time capacity.

A single delivery contains different types of load carriers

A wide range of materials is required for the manufacturing process of a module. These materials differ in size and weight and require different load carriers. For instance, small materials are put into a crate that is placed on a kitkar and large or heavy materials are placed on a pallet. These large and heavy materials can often not be handled by hand and require tools such as cranes to lift them. The tools to facilitate these handling procedures are not present in the standard unpacking area where the crates with small materials are delivered to but are present in the cleaning area. Right now, materials with different types of load carriers are allocated to a single processing step. When a set of materials is picked in the warehouse, some materials are brought to the unpacking area while they should be delivered at the cleaning area. Or materials are delivered to the cleaning area without any documentation (picking list or order rule or any other form of identification). Also, there currently is no capacity reserved for the unpacking and cleaning of materials in the cleaning area. These problems cause WIP inventory in the cleaning and unpacking area and therefore impact the flow of materials to the production area in terms of capacity and handling issues.

Materials are allocated in the wrong processing steps

A core problem related to high WIP inventories is the fact that some materials are allocated to the wrong processing steps. Currently, the materials are allocated to processing steps forming a set of materials. A single set of materials is delivered at once to the production area. Due to an incorrect allocation of materials to processing steps, materials are supplied to the production area while they are not yet needed in production, causing the materials to be idle inventory. This wrong allocation of materials can also influence the WIP inventories at the cleaning and unpacking area. These materials should not be cleaned or unpacked at that moment and in cases where there is no capacity reserved or available it causes the WIP inventories to be higher.

There is no feedback to the ERP-System
Currently, when an order is placed by the customer, Systems-2 gets a delivery date for this order from the customer. This future delivery date is inserted in the ERP-system called BAAN. From this future delivery date BAAN sets up several milestones between the delivery and starting date. These milestones are guidelines for certain processing steps to ensure that the module is manufactured in time. However, in reality it can occur that a milestone is not met. This can be caused by several reasons, such as waiting for material, the burnout ovens are occupied, or the required tooling is not present. When this happens, the milestone might be met at a later point. This however is not communicated back to BAAN. So, after the first deviation in the manufacturing process, the production planning set up by BAAN, is not rescheduled making it redundant. Due to the inability of rescheduling in BAAN, there is no up-to-date production planning that forces the logistic department to plan the production of all modules through verbal communication. This way of working is prone to making mistakes due to having to make ad-hoc decisions for the production and material planning while not having a good overview of the complete production and material planning resulting in high WIP inventories at the production, cleaning and unpacking area.

To allocate correct materials into the correct processing steps and provide quality feedback to BAAN, first information about how these processing steps should be designed must be researched. Reallocating materials in processing steps will not solve the problem if there are still too many materials delivered at once causing high WIP inventory levels and capacity issues at the unpacking process. Similar reasons will cause that feedback to BAAN will also not solve the problem directly. Therefore, the selected core problems for this research are:

1. Too many materials are delivered at once
2. A single delivery set contains different types of load carriers

By first addressing these two problems the foundation for tackling the other two core problems is established.

1.4 Problem approach

In an ideal situation Systems-2 wants to control the WIP inventory at the production, cleaning and unpacking area such that the number of modules manufactured can be higher in the future. To achieve this, a decision must be made on the quantity and timing of materials that need to be delivered to the production area. This decision must be made because the delivery of materials requires time due to the unpacking process, making a just-in-time delivery not applicable to this situation. The problem at Systems-2 can be seen as a lot sizing problem which aims to specify when to have material delivered or produced, as well as quantities required (Jeunet & Jonard, 2000).

For this research, a model is developed that determines the optimal number of materials to deliver at what moment. First, knowledge is gathered about how the existing process, encompassing the flow of materials to the production area is currently working. Second, data is collected regarding the materials required to produce the modules is needed. This includes information about the types of load carriers that are used for the materials. Third, data about the capacity of the unpacking area and the cleaning area is gathered. This knowledge serves as input for the model. The model's solutions are then presented, analyzed and evaluated. Finally, the implementation of the model is discussed by addressing the steps required to implement the model.

The stakeholders in this research are the departments: Management, Production, Logistics, Factory Engineers, and Quality. Every module has a team that consists of a project leader, factory engineers and mechanics. For every module a factory engineer and mechanic is selected as a representative for the module. These representatives are consulted and kept informed. The Quality department won't have much influence in this research but is taken into account since cleanliness and quality is an important aspect in the production process. The Logistics department is the key stakeholder and is therefore managed closely. The result of this research is primarily of their interest. Therefore, every week, one mandatory and one optional meeting is scheduled to discuss the progress and direction of this research. The management of Systems-2 must also be informed, this is done by a monthly meeting in which an update will be given about the research.

A potential risk for this research is the selection of software that is used to develop the model. The software required for solving such problems can cause for high investment costs. For Systems-2 it is essential that the model can be used after completing this research. This means that it is preferred to use a software that is accessible for Systems-2. This risk is considered during this research.

1.5 Research questions

The scope of this research is limited to the Systems-2 cleanroom customer-specific parts, meaning that potential floor stock is excluded. Also optimization of the supply of materials from external suppliers is not part of the scope for this research. The goal of this research is to control the WIP inventory at the production, cleaning and unpacking area. To achieve this goal, a research is conducted on how a model can be created and implemented that takes supply capacity, batching and consolidation of materials into account and uses this information to calculate the number of materials that should be brought to the production area at what time. This research objective leads to the following research question that needs to be solved:

How can the WIP inventory at the production, cleaning and unpacking area be controlled while taking maximum capacity at the cleaning and unpacking area, and different types of load carriers into account?

This main research question is divided into sub-questions. These sub-questions are structured in 4 phases and every sub question represents a chapter in this research. The information gained from answering the sub-questions is used to answer the main research question.

Current system analysis

The first step of this research is to get a better understanding of the problem and an overview of the existing processes corresponding to the problem. This done in Chapter 2 by answering the first research question:

1. What do the processes regarding the current material and production planning look like?
    a. What is the current production process?
    b. What is the current material and production planning process?
    c. How are the materials delivered towards the production area?
    d. What type of materials need to be delivered to the production area? And how can they be categorized?
    e. How is the demand of materials distributed per module?

In Chapter 2, the current system analysis is presented by first describing the current production process. First, the layout of the production area including the unpacking and cleaning area is provided as well as the current material and production planning process. Second, we zoom in on the delivery process. Finally, we categorize the types of materials to get insights in the load carriers that are used.

Literature search

Once the current system is analyzed, we turn to the literature to find ways to tackle our problem/ This is done in Chapter 3 by answering the second research question:

2. What methods are proposed in the literature to control WIP inventory at the production area?
   a. What is the current state-of-the-art of material and production planning methods?
   b. What solution approaches are proposed in the literature?
   c. What are relevant Key Performance Indicators (KPIs) to measure the performance of production planning methods?

In Chapter 3, a literature review is performed to get an overview of the current state-of-the-art of material and production planning methods. In addition to that, the existing literature is reviewed to search for possible solution approaches that are applicable for this research and how these solutions can best be measured.

Model design

After consulting the literature, we create the model that is able to find a new setup policy. This is done in Chapter 4 by answering the third research question:

3. What should the model for minimizing WIP inventory while improving the flow of materials look like?
   a. What is input data is needed for the model?
   b. What are the restrictions for the model?
   c. What is the best suitable solution approach for this problem?

Based on the literature review, an MIP model is developed. The knowledge gathered from the existing literature regarding material planning and production methods is synthesized to develop a model to control the WIP inventory at the production, cleaning and unpacking area. In this phase also the input data, restrictions and solution approach for this model are specified.

Analysis of results

Once the model has been developed and created, the next step is to apply the model for this research. In Chapter 5, the model is solved to generate a new setup policy. We get to know what policy Systems-2 could use to reduce the WIP inventory at the production, unpacking and cleaning area. This is done by answering the fourth research question:

4. How can a policy reduce the WIP inventory at the production area?
   a. What is the best configuration of the model?
   b. What are the results compared to the current situation?
   c. What is the impact of varying input parameters on the models outcomes?

First, a numerical analysis is performed to find the best performing configuration for the model. Once the best configuration is established, the model is applied to two modules resulting in a setup policy for each module. This setup policy is then compared to the current setup policy to study the impact of the new setup policy. Finally, changes in the key parameters are applied to find out if the model output is valid and reliable.

Implementation

The final stage of this research, the necessary steps for the implementation is discussed. The full implementation of the model is not included in the scope of this research. Therefore, in Chapter 6 an implementation plan is constructed by answering the following research question:

5. How can the solution be implemented and used by Systems-2?
   a. What further steps are required to implement the model and its results?
   b. How can the model be used by Systems-2?

First, the necessary steps for a functional implementation of the models output are described to serve as a guidance for a successful implementation. In addition to that, it is important to determine and show how the model can be used by Systems-2 such that the module can be applied to the remaining modules.

2 Current System Analysis

In this Chapter, the current situation regarding the material and production planning process will be described. The first research question: "What do the processes regarding the current material and production planning look like?" is answered. Section 2.1 presents the layout of the production area and how the current material and production planning process is structured. Section 2.2 describes the processes that take place in the unpacking area and the cleaning area. Section 2.3 presents the categorization of materials and how materials are allocated to the current processing steps. Section 2.4 concludes this chapter.

2.1 Material and production planning process

This paragraph describes how the current material planning and production planning process is working. First the production area is described. Second, the routing of a module is elaborated followed by a description of the current production and material planning process.

2.1.1 Production area

As mentioned, the production of the modules that Systems-2 produces takes place in a cleanroom environment. A cleanroom is a self-contained space in which a maximum number of particles per cubic meter are allowed. Since even the smallest particles can cause defects in the final product in the semiconductor industry, cleanliness is an import aspect. The materials that are delivered in to the cleanroom for production must therefore follow a process such that the materials do not contain as little particles as possible. This is a time-consuming process that takes place in the unpacking area and/or Large & Medium cleaning area.

In Figure 5, a simplified version of the lay-out of the production area, including the unpacking area and the Large & Medium cleaning area is shown. Both the unpacking area and Large & Medium cleaning consist of three compartments due to the cleanliness restrictions. In the production area it can be seen that every module that is produced has its own dedicated production area indicated by M1, M2, M3A, M3B and M4. Once the production of a module is fully completed it leaves the production area.

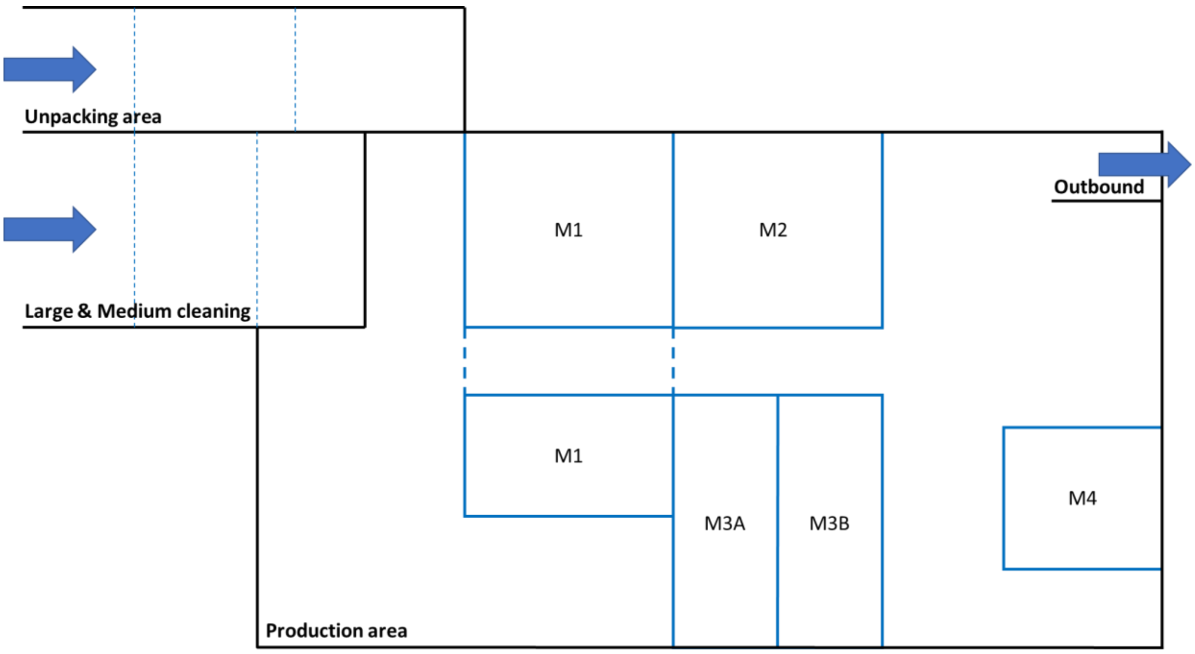


Figure 5: Simplified layout of the production, unpacking and cleaning area

The cleaning and unpacking process starts in a conventional space with no requirements which is the first compartment in Figure 5 (from left to right). Here the first cleaning or unpacking steps take place and once ready, the material in question is transported to the second compartment. This compartment complies with ISO 7 norms, meaning that a maximum number of 2390 particles larger than 5 µm per cubic meter are allowed. Here again, the required cleaning and unpacking steps take place before the material is transported to the final compartment that is compliant with ISO 6 norms ready to enter the production area.

2.1.2 Routing

For every module a routing has been created. Every module is divided into several processing steps specified by a number. It is the route that a module follows to complete the manufacturing, hence the name 'routing'. In this research, when we talk about 'routing' we talk about the routing of a module that exists in Baan and not a physical route. The routing is made up of so called processing steps. Every processing step is characterized by its own step number, step description, department, required machine, processing time in minutes and hours. Figure 6 shows the first page of the routing as it is presented in Baan. In the first column under 'Bew.', the step numbers of the processing step are shown in an ascending order starting at processing step 10.

Bew.	Taak	Omschrijving	Afd.	Omschrijving	Krit cap.	Srt. bew.	Bewrk. gereed	Tel-punt
10	8992	[redacted]	798	[redacted]	Manc		Ja	Nee
11	8188	[redacted]	700	[redacted]	Manc		Nee	Nee
15	8184	[redacted]	710	[redacted]	Mach		Nee	Nee
20	8184	[redacted]	710	[redacted]	Mach		Nee	Nee
30	9256	[redacted]	700	[redacted]	Manc		Nee	Nee
31	6301	[redacted]	700	[redacted]	Manc		Nee	Nee
32	9990	[redacted]	999	[redacted]	Manc		Nee	Nee

Figure 6: Module routing in Baan

In the routing, there are processing steps for washing, inspection but also for assembling parts to the module for example. These assembling steps typically require materials that need to be assembled, apart from the assembling steps that consist of assembling sub-assembly. Most of the existing processing steps have its own set of materials that need to be available in at the production area for assembly. There is no fixed minimum of maximum number of materials that is assigned to a processing step. Instead, every processing step in the routing has its own work instructions developed by the factory engineers of this specific module. So, the number of materials that are assigned to a processing step is dependent on how the work instructions (WIs) are designed for a processing step as seen in Figure 7.

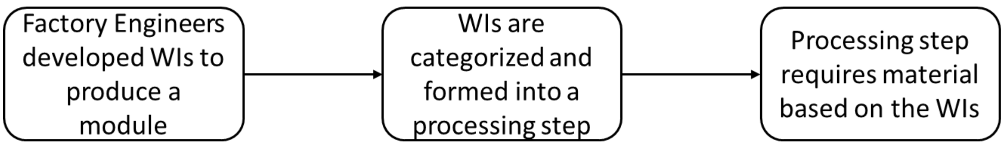


Figure 7: Design of processing steps

By having developed a routing for every module, it might seem that it is easy to predict what the TT of a module is and what materials are needed at a certain point in time. However, there are steps in the production process that every module must undergo. These steps include cleaning, dry cleaning, leak testing, inspection, measuring and packaging & shipping. These steps cannot always be performed simultaneously and can cause that a module must wait until the module that occupies the facilities for this step has completed it. In this time the production process for the waiting module is stopped. This waiting time is taken into account, by including buffer times in the routing. These buffer times cause uncertainty in the actual starting moments of processing steps in the production process and can vary from 2 to 7 days depending on the type of processing step a module must undergo. Leading up to a total of 26 days of buffer time for a single module. To get an idea, a certain module takes 51 days to complete excluding the buffer times and 77 days including the buffer times leading up to an increase of almost 51%. Figure 8 visualizes the impact of buffer times on the planning of the production process. Having these buffer times makes it hard to predict when material demand occurs for a processing step during the manufacturing process of a module. This uncertain demand for materials is considered during the development of the model.

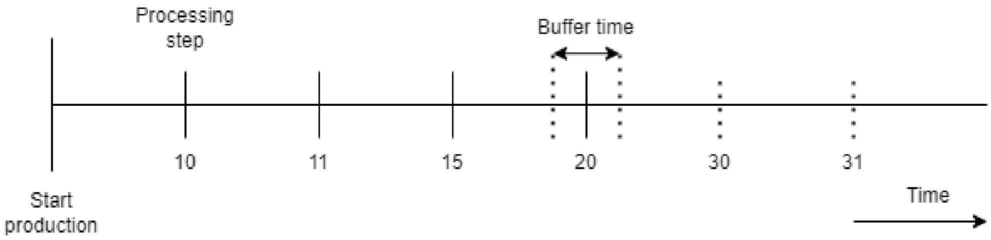


Figure 8: Visualization of the impact of buffer times on the planning of processing steps

2.1.3 Production planning

The material planning and production process starts with an order from the customer. The customer issues a desired delivery date for this order. This order is then registered in BAAN where a starting date for the project is calculated. The calculated starting date in BAAN refers to the date at which the module is planned to be cleaned before entering the cleanroom.

Every module has its own routing. Once a project is started, the routing is used to plan the project and monitor the process. As the project progresses, new process steps that require materials become available in BAAN, meaning that the Production Assistant (PA) is allowed to start the next processing step. However, in practice, BAAN is not leading in the production planning process. The reason for this is that the progress of the actual production is not communicated back to Baan. So, as the production process progresses, BAAN does not update planning. Instead, the progress of the production is manually registered in an Excel overview and is called: the MAS. This is an abbreviation of the Dutch words: Montage Afloop Schema which can best be translated to: Assembly Completion Schedule. In this MAS the routing of a module is presented in a chronological order with lines displaying the connections between processing steps.

Every day, a meeting takes place called a Whiteboard meeting. For every module, fifteen minutes are scheduled to discuss the progress of the manufacturing of this module. This meeting is attended by a quality engineer, quality inspector, head mechanic, and project leader of the corresponding module, and the PA and production leader. In the MAS, the mechanics evaluate each processing step by indicating when a process step is planned to start, when it is being worked on, when it is completed and if a problem occurs. This way the progress of a module is registered. The information discussed during the Whiteboard meeting regarding the processing steps is then used as input for the call-off procedure of the Production Assistant (PA). BAAN has a connection with Excel via IQBS. This connection enables to export and view information from BAAN in Excel. In Excel the actual process of calling-off processing steps takes place, this is done by the PA of Systems-2. Based on the routing created in BAAN, the order line(s) are presented that are allowed to be called off by the PA. If during the meeting it becomes clear that a certain processing step can be started and the order line(s) are presented in Excel, the PA then sends an e-mail with the order line(s) that must be picked by the warehouse.

Besides the order lines, Excel is also used the planning of other processing steps such as washing, dry cleaning, leak testing and inspection. Here the order planner must make a decision on which module is prioritized for undergoing one of the above mentioned process steps. A prioritization is and can only be made when there is a queue. If there is no queue it is follows the first come first serve rule. Figure 9 shows a global overview of the production planning process.

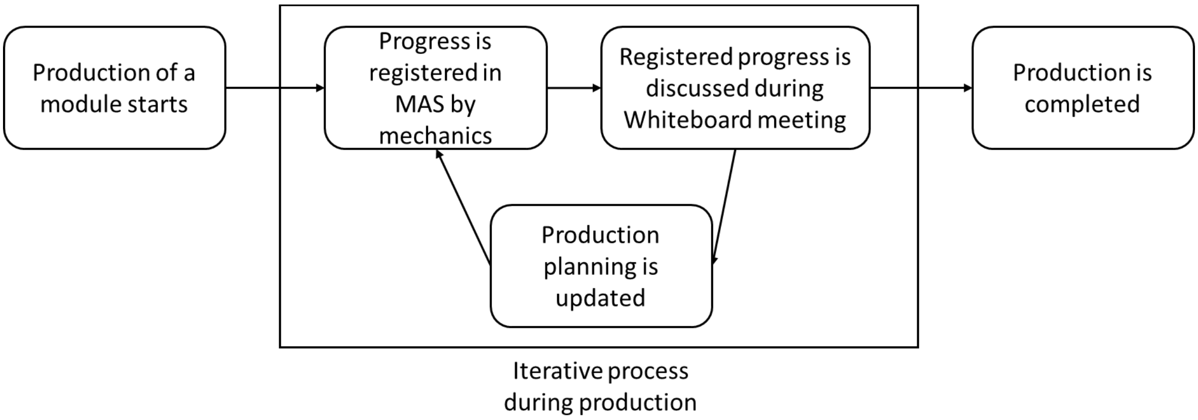


Figure 9: Global production planning process

The information regarding the current production, routing and the production planning process serves as relevant context for this research. It shows that the production process at Systems-2 is not a standard process, and it displays that due to the cleanliness restrictions, Systems-2 can be seen as its own supplier of materials to the production area. This paragraph also describes how the routing of a module is designed and how it leads to the use of processing steps and the materials allocated to it, which will be further elaborated in Section 2.3.2. The current planning process also shows that most of the planning is done through verbal communication and that it is based on the routing and its processing steps. So the actual moment material must be delivered to the production area is dependent on the status of the MAS and the discussions during Whiteboard meetings which causes uncertainties in the timing of material demand causing possible WIP inventory and capacity issues.

2.2 Delivery processes

This paragraph will describe the physical delivery process starting from the moment materials from external suppliers arrive at VDL ETGA. From there the materials are stored temporary after which they must be delivered to and into the cleanroom.

2.2.1 External supply of materials and storage

Some materials of the modules are manufactured internally and some materials are purchased by external suppliers. These purchased parts are delivered to the warehouse of VDL ETGA. Here the materials are temporary stored before being brought to the production site. Depending on their weight and volume, the materials are either stored in a mini-load storage, vertical carousel storage system or pallet storage. The arrival of materials from external suppliers depends on the routing of the modules. In general, the materials are scheduled to arrive ten days before they are needed in production. Unlike in the production planning, here the routing in BAAN is used to indicate when certain materials are needed in production.

2.2.2 Unpacking

The materials required for production are now stored inside the warehouse at VDL ETGA in a mini-load or pallet storage. The delivery of materials to and into the cleanroom starts when the PA calls-off an order. This is done by sending an e-mail to the warehouse. From there the warehouse employees picks the requested order. First a picklist is printed and a kitkar with blue crates is placed at the order pick place in front of the mini-load. Once the order is picked, the warehouse employee transports the materials to the storage location near the cleanroom in a kitkar. This is material handling equipment with 8 shelfs that can carry three crates per shelf. This process, starting from the moment an order is called off by the PA until the moment that the materials are delivered to the unpacking area, has a uncertain lead-time that varies between two to four days.

All these materials are cleaned at the supplier and are packed in three layers of plastic and require an unpacking procedure. This unpacking procedure begins by bringing a kitkar with blue crates into the unpacking room. Here the unpacker takes out a single blue crate filled with materials. The first layer of plastic layer is removed and the material is placed inside a green crate indicating that the first layer of plastic is removed. This is done for all materials. The kitkar with green crates is then transported into the unpacking area that is compliant with ISO 7 norms. Here the unpacker follows the same process by grabbing a green crate with materials, unpacking these materials and placing them inside a yellow crate indicating that two layers of plastic have been removed. Once this is done for all materials, there is one clean plastic layer left and the kitkar is ready to enter the cleanroom compliant with ISO 6 norms.

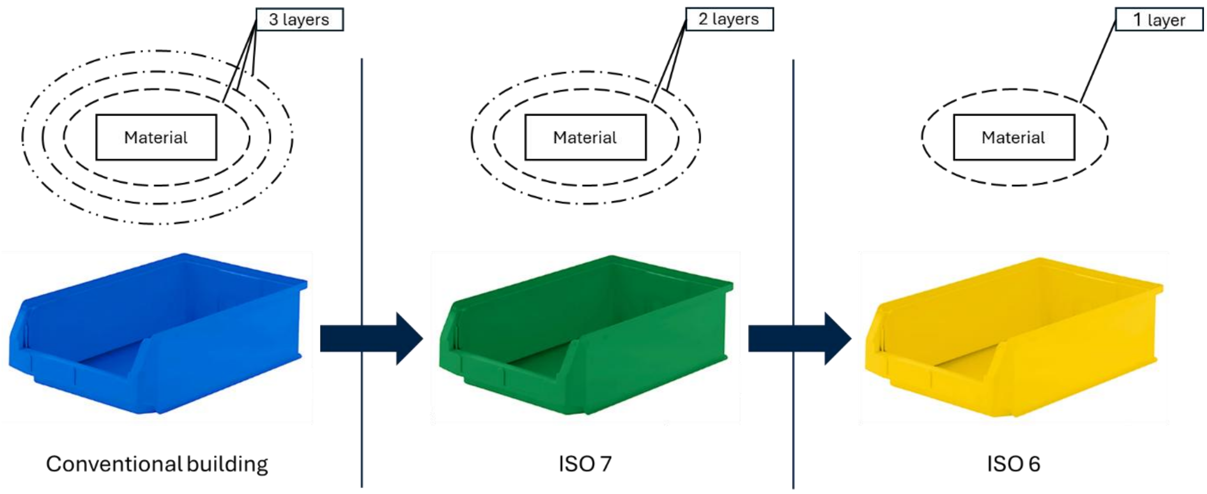


Figure 10: Visualization of the unpacking process

The material must be packed in one layer of plastic when entering the cleanroom, because there are still particles in the cleanroom that will eventually drop on the material. The last layer of plastic protects the material for these particles to ensure that it is still clean when it is ready to be assembled.

Internal calculations show that the required time to unpack one layer for a single material is two minutes. This includes additional tasks such as counting the number of materials and the administration on the picking list, and the handling of the crates. Since there are two unpacking moments, it would take four minutes to complete the full unpacking procedure for a single material. However due to not having to perform all of the additional tasks mentioned above, experience has learned that the second unpacking moment takes less time. Later in this report it can be seen that the number of materials required for a module range between 450 and 2287. Due to the processing time and the high number of materials that pass the process, this activity requires planning.

2.2.3 Large and medium cleaning

Some of the materials that are used in the manufacturing process cannot be handled by hand due to their shape and/or weight. These materials must not be brought to the same location as the kitkar storage but must be brought to the large and medium cleaning area. This cleaning area is designed to handle large materials. It consists of three compartments: an arrival area, a wet cleaning area and a drying and particle inspection area compliant with respectively ISO 8, ISO 7 and ISO 6 norms. This way the larger materials follow a similar way of entering the cleanroom compared to the materials that need to be unpacked. These compartments have more space and are equipped with the necessary facilities such as cranes and hoisting tools to handle these larger materials. The materials that cannot be handled by hand and come from external suppliers are packed in three layers of plastic similar to the small materials. Here in the first two compartments a layer of plastic is removed such that one layer remains before entering the cleanroom. As said in the problem description, even though the required facilities are present in the cleaning area, there is currently no capacity reserved for unpacking materials in the cleaning area.

In this paragraph the unpacking process at the unpacking and cleaning area is shown. It shows that the supply of materials to the production area is not a standard process and that it requires a significant amount of time to deliver an item to the production area making a just-in-time delivery not possible. Also, due to the cleanliness restrictions, the employees must change clothes which takes time. Currently there is one Full-time equivalent (FTE) available for the unpacking and cleaning of all the materials. This means that there is a limited capacity in terms of time for the unpacking of materials in both the unpacking and cleaning area. This is something that should be considered when creating a model for this problem.

2.3 Material types

This paragraph will describe what sort of materials and load carriers of materials are used for the manufacturing of the modules. Also, the way demand arises for certain materials will be discussed. Up until this point four modules are mentioned. Due to the way Module 3 is assembled and the size compared to other modules, Module 3 is from a logistical point of view divided into two modules: Module 3A and Module 3B. In the coming sections we view Module 3 as Module 3A and Module 3B.

2.3.1 Categories

Every module consists of a certain composition of materials. The modules differ in size and number of materials required for production and consist of 838, 1072, 352, 486, and 229 Stock Keeping Units (SKUs) for respectively Module 1, 2, 3, 4, and 5. Some of these SKUs are used across multiple modules and are considered floor stock, which are excluded from the scope of this research. The SKUs all have their own characteristics; however, it is possible to categorize these SKUs in terms of size and weight which determine the load carrier of the material. For systems-2 four categories can be recognized:

1. SKUs that fit in a crate of 30x52x45 (Kitkar crate)
2. SKUs that fit on a euro pallet 120 x 80 (Pallet)
3. SKUs that are delivered in tailor made pipe carriers (Pipe carriers)
4. SKUs that must be delivered through the cleaning area (special)

For this research it is important to categorize the materials that are used for the production of all the modules. By categorizing the materials and the number of materials per category, we can get an idea how many materials follow the unpacking process through the unpacking and cleaning area. This can later be transformed into input for modelling the capacity in the model. In Figure 11 the number of materials per load carrier per module is shown. Here it is important to note that these are not the SKUs, but count of materials, one SKU for instance can occur ten times in a module. Most materials of every module, roughly 88% or more, can be transported to the production area through kitkar crates.

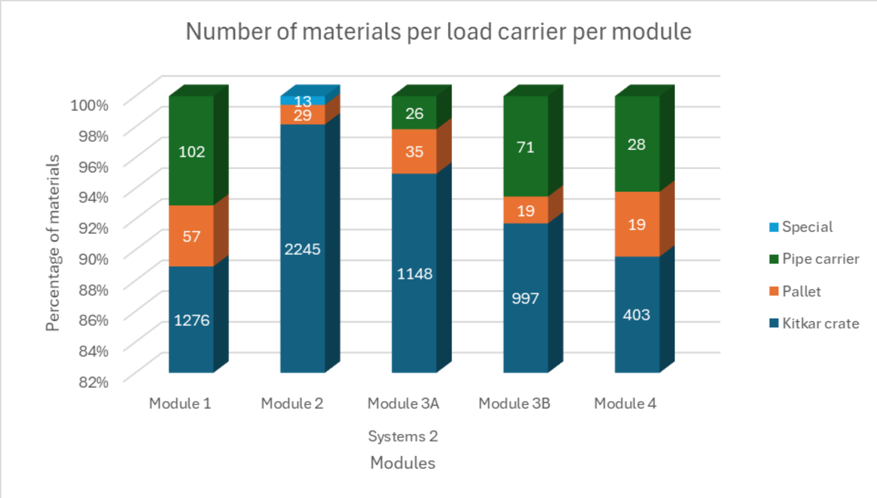


Figure 11: Number of materials per load carrier per module

2.3.2 Materials allocated to processing steps

The demand of materials for a certain module depends on how many materials are allocated to a single processing step and the moment of calling-off a certain processing step. In this sub-paragraph we will take a look at the allocation of material across the processing steps per module and how this relates to problems associated with the delivery process and the high WIP inventory levels. The following five Figures show the allocation of materials across processing steps per module.

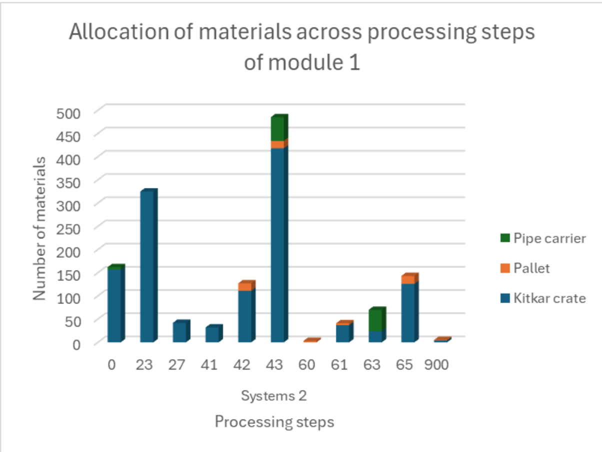


Figure 12: Material allocation of module 1

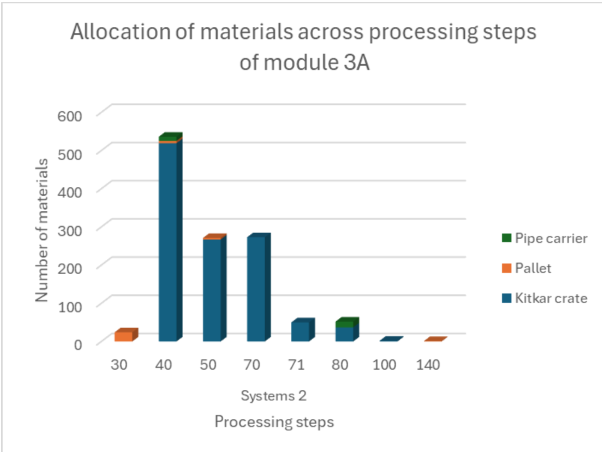


Figure 14: Material allocation of module 3A

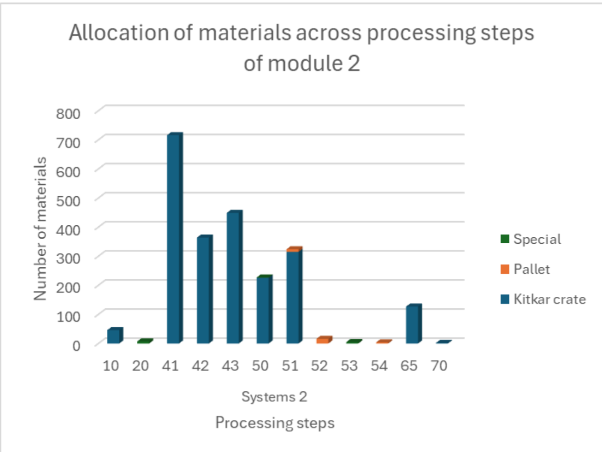


Figure 13: Material allocation of Module 2

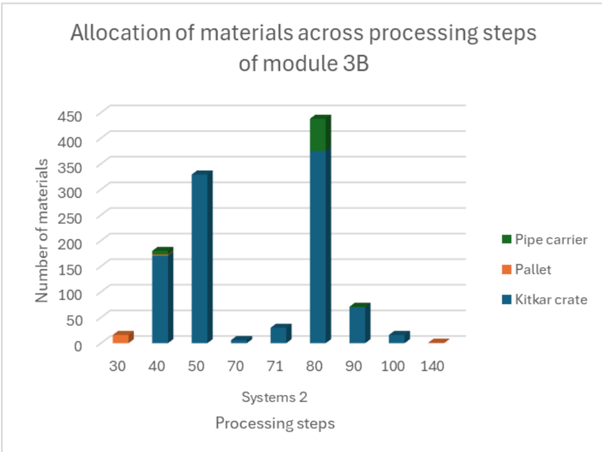


Figure 15: Material allocation of Module 3B

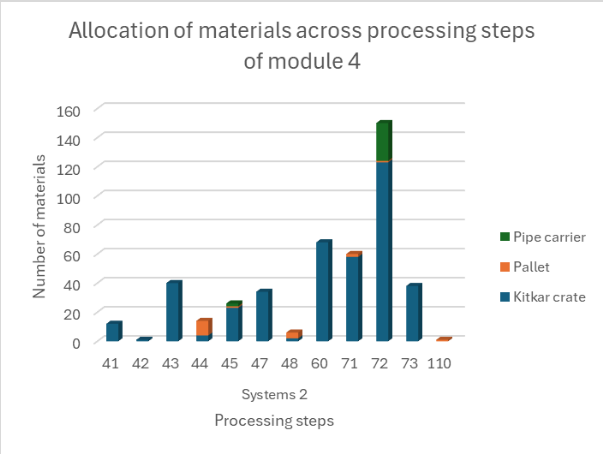


Figure 16: Material allocation of Module 4

In the above presented Figures, the number of materials is presented on the y-axis and the processing steps identified by the numbers on the x-axis. There are high fluctuations in number of materials for every module. Some processing steps contain a manageable number of materials that do not cause problems for Systems-2. There is enough capacity in the unpacking area, and there is little to no inventory inside the cleanroom when such a processing step is called-off. Conversely there are processing steps that contain a large number of materials causing for a peak in the workload in the unpacking area and causing high inventory levels inside the cleanroom. Also, it can be seen that some processing steps contain multiple load carriers that cause problems in terms of time capacity for the cleaning area. Ideally this workload is spread evenly over the processing steps such that there are no capacity issues for the unpacking process and there is a lower WIP in the production, cleaning and unpacking area.

2.4 Conclusion

The way that the routing is developed in combination with the current production planning process, shows that verbal communication is needed in order to actually plan the delivery of materials to the production area. It can also be seen that this is not a standard process and a just-in-time delivery is not possible due to the cleanliness restrictions. To deliver materials into the cleanroom, the materials either need the be unpacked or cleaned. This is done in the unpacking and cleaning area. For the unpacking and cleaning, there is currently one FTE available. To reduce the WIP inventory in the unpacking and cleaning area, we must allow a maximum number of materials to be delivered to the production area such that one FTE is able to unpack and clean all the materials before new materials need to be delivered. The current material allocation also shows that there are certain processing steps that have a relatively high number of materials that need to be delivered to the production at once. The current situation shows that there is a need for a new configuration of the processing steps such that there are no capacity issues related to the unpacking process and such that the WIP inventory inside the cleanroom is minimized.

3 Literature study

In this Chapter, a literature study is performed. The literature search approach is described in section 3.1. In Section 3.2 the state-of-the-art on lot sizing models is presented followed by possible solution approaches for this problem in Section 3.3. Section 3.4 describes how stochasticity can be included in this research. Section 3.5 describes the relevant Key Performance Indicators for this problem The findings during this literature study will be used as input for designing a model in Chapter 4. This chapter answers the following research question:

What methods are proposed in literature to control WIP inventory?

3.1 Approach

The current situation at VDL ETGA is studied and we can explore the existing literature in this field. To make sure that the search for literature is done efficiently, first the terminology in this field is explored. During this process, key-words can be selected that are ought to be relevant. These key-words can then be used to form accurate search strings for this research.

3.1.1 Terminology

Since the publishing of Ford Whitman Harris'(1913) seminal paper, the lot sizing problem, which aims at determining economic (production or order) lot sizes by balancing inventory and setup or order costs, has received wide attention in both the academic literature and in practice (Glock & Grosse, 2014). Karim et al. (2003) state that the lot sizing problem is one of the most important and also one of the most difficult problems in production planning. Due to its wide attention and importance, many variants of the lot sizing problems have emerged that can be applied to specific situations. For this research we deal with uncertainty in both supply and demand, capacity restrictions, and various load carriers. However, the terminology for these considerations must be used correctly. Karim et al. (2003) describes eight characteristics that affect classifying, modelling and the complexity of lot sizing decisions. Jans & Degraeve (2006) give an overview of recent developments in the field of modeling deterministic single-level dynamic lot sizing problems. Their first line of research focuses on modeling the operational aspects in more detail. And their discussion is organized around five characteristics. The characteristics are summarized in Table 1.

Table 1: Characteristics that affect the classification of the lot sizing model

Karim et al. (2003)	**Jans & Degraeve** (2006)
Planning horizon	The set up
Number of levels	Characteristics of production process
Number of products	Inventory
Capacity or resource constraints	Demand
Deterioration of items	Rolling horizon
Demand	
Setup structure	
Inventory shortage	

Even though Jans & Degraeve (2006) mention less characteristics, there are still a lot of similarities between the characteristics that are mentioned. Also, Jans & Degraeve (2006) focuses on the modeling of deterministic single level dynamic lot sizing problems. They have already incorporated some characteristics in their scope instead of focusing on the lot sizing problem in general. Karim et al. (2003) focuses on a more general overview of the lot sizing problem, and elaborates every characteristic explicitly. By characterizing the lot-sizing problem according to the definitions given by Karimi et al (2003), we can reduce the number of papers that ought to be relevant for this research by excluding the papers that are not representative of the situation. Therefore in this research, the terminology regarding for formulating accurate search strings is selected according to the characteristics given by Karim et al. (2003). The characteristics and how they do or do not relate to the problem are presented below.

Planning horizon
The planning horizon is the time interval on which the master production schedule extends into the future. The planning horizon may be *finite* or *infinite*. A finite-planning horizon is usually accompanied by dynamic demand and an infinite planning horizon by stationary demand. Another variant of the planning horizon is a *rolling horizon* which is usually considered when there is uncertainty in data. Under this assumption, optimal approaches for each horizon act as heuristics but cannot guarantee the optimal solution. Since we are dealing with dynamic demand we use a finite-planning horizon

Number of levels
Production systems may be a *single-level* or *multi-level*. In single-level systems, usually the final product is simple. The end item is directly produced from raw materials or purchased materials with no intermediate subassemblies. Product demands are assessed directly from customer orders or market forecasts. In multi-level systems, there is a parent-component relationship among items. The output of an operation (level) is the input for another operation. At systems-2 there is this parent component-relationship thus we speak of a multi-level system.

Number of products
There are two principal types of production system in terms of number of products. In *single-item* production planning, there is only one item for which the planning activity must be planned, while in *multi-item* production planning there are multiple items. In the production area there are dedicated workspaces for the four modules. In this research the focus is on controlling the WIP inventory for each single module separately. But every module consists of multiple items, meaning that we deal with a multi-item system.

Capacity or resource constraints
Resources or capacities in production systems include manpower, equipment, machines, budget, etc. When there is no restriction on resources. The problem is said to be *incapacitated*, and when capacity constraints are explicitly stated, the problem is named *capacitated*. This research is capacitated as there is a maximum capacity regarding the supply of materials to the production area.

Deterioration of items
In the case that deterioration of items is possible, we encounter restrictions in the inventory holding time. In this research items do not deteriorate, and therefore not included in the search string.

Demand
Demand type is considered as an input to the model of the problem. *Static demand* means that its amount does not change over time, it is stationary or even constant. While *dynamic demand* means that its amount changes over time. If the value is known in advance (static or dynamic), it is termed *deterministic*, but if it is not known exactly and the demand values occurring are based on some probabilities, then it is termed *probabilistic*. In addition, if the production system is single-level, then we talk about *independent demand*. If the production system is multi-level, then we talk about *dependent demand* as the demand depends on the demand of its parents' level. Due to the uncertainties in the production process, we speak of dynamic demand. So the timing of the demand can change, but the material types and number of materials per material types are known beforehand meaning that the quantity of demand is deterministic. Since there is a parent-component structure as described in the number of levels, demand is dependent.

Setup structure
There are two types of setup structure: *simple setup* structure and *complex setup* structure. Where in a simple setup structure the setup time and cost are independent of the sequence and decisions in previous periods. But when the setup time and cost are dependent on the sequence and decisions in previous periods, it is termed a complex setup structure. A complex situation is mostly present in a production setting with changeover between different products that incur setup time and setup costs. For this research we are dealing with a simple setup structure. We consider the time of the unpacking process as the setup time. The unpacking time is not dependent on a sequence or previous decisions as there are no changes required in the process for certain materials that influence the setup times.

Inventory shortage
If shortage is allowed it means that it is possible to satisfy the demand of the current period in future periods (*backlogging* case), or it may be allowable for demand not to be satisfied at all (*lost sales* case). In practice shortages can and do occur due several reasons such as quality issues of material or late delivery of suppliers. However, this is not something that is allowed and considered in the planning of this production and material production process. So, backlogging is not included.

3.1.2 Search strategy
By characterizing this research according to the above mentioned eight characteristics makes it possible to search more accurately in the existing literature. The Scopus database is used to search for relevant literature. In this database one must formulate so called search strings. It is plausible that there is not a paper that matches the exact characteristics of this research. Therefore, multiple search strings are formulated with different compositions of the characteristics described in Section 3.1.1. The search strings are presented in Table 2.

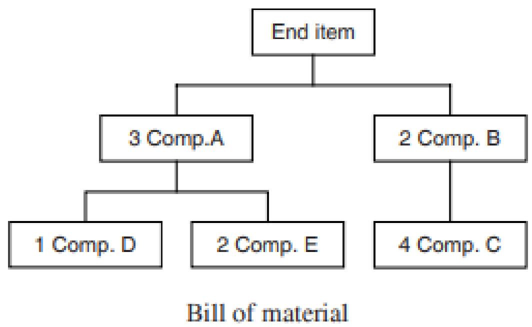
Table 2: Search strings used for Scopus database

Search string	Documents	Relevant
TITLE-ABS-KEY ("Lot-sizing" OR "Lot-size" OR "Lot sizing" OR "Lot size" AND "Capacitated" AND "Dynamic")	263	8
TITLE-ABS-KEY ("Lot-sizing" OR "Lot-size" OR "Lot sizing" OR "Lot size" AND "Capacitated" AND "Dynamic" AND "Multi-Item" OR "Multi item")	65	8
TITLE-ABS-KEY ("Lot-sizing" OR "Lot-size" OR "Lot sizing" OR "Lot size" AND "Capacitated" AND "Dynamic" AND "Multi-Level" OR "Multi Level")	28	4
TITLE-ABS-KEY ("Lot-sizing" OR "Lot-size" OR "Lot sizing" OR "Lot size" AND "Capacitated" AND "Dynamic" AND "Multi-Level" OR "Multi Level" AND "Multi-Item" OR "Multi Item")	6	4
TITLE-ABS-KEY ("Lot-sizing" OR "Lot-size" OR "Lot sizing" OR "Lot size" AND "Capacitated" AND "Dynamic" AND "finite" AND "horizon")	17	2

To get a better understanding of the problem at the start of this research, some literature already has been searched for. During this search some papers deemed relevant and will also be used during this research if needed.

3.2 Findings
Research on lot-sizing dates to the early twentieth century, and a large number of different lot-sizing problems have been identified, for which an even larger number of modeling approaches and algorithms have been developed Buschkühl et al. (2008). The research on lot-sizing started with the classical Economic Order Quantity (EOQ) model, which assumes deterministic static demand, continuous time, and an unlimited replenishment lot-size. The objective is to minimize the sum of

ordering and inventory holding costs Buschkühl et al. (2008). When the assumption of the steady-state demand rate is dropped – i.e., when the amounts demanded in each period are known but are different – and furthermore, when inventory costs vary from period to period, Wagner & Within (1958) present the dynamic version of the model. Over the years many model variations and extensions have been developed by various researchers to make the lot-sizing applicable to industry-specific situations of which one of them is the dynamic Multi-Level Capacitated Lot-Sizing Problem (MLCLSP) introduced by Bellington et al. (1983). The main idea of this formalization is to link end items demand with internal components needs thanks to a matrix called "Gozinto". The latter is an algebraic translation of the bill-of-material (BOM) Comelli et al. (2008) as shown in Figure 17.



$$
\begin{array}{c}
\\ \text{E. I.} \\ \text{A} \\ \text{B} \\ \text{C} \\ \text{D} \\ \text{E}
\end{array}
\begin{array}{c}
\begin{array}{cccccc} \text{E. I.} & \text{A} & \text{B} & \text{C} & \text{D} & \text{E} \end{array} \\
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 0 \\
2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 4 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 0
\end{pmatrix}
\end{array}
$$

Gozinto Matrix

Figure 17: A bill of material and its representation thanks to Gozinto matrix Comelli et al. (2008, p. 214)

The complexity of such a matrix depends on structures of BOM. Comelli et al. (2008) distinguish three structure's types: Serial (or linear), Assembly and General. Besides these three structure's types Bellington et al. (1983) includes a parallel structure. The series structure illustrates a single product produced in a series of steps. The parallel structure illustrates a structure where several items proceed through the same production steps. The assembly structure represents a product made by a complex assembly process with every assembly or material has exactly one successor. The general product structure depicts commonality of components, the situation where components or materials are used in more than one successor item. This is the general form of a material requirement problem. To take full advantage of commonality, a list of successors (also parents or user-items) of each component must be maintained. The logical place for this information is in the BOM describing the multiple-use item Bellington et al. (1983).

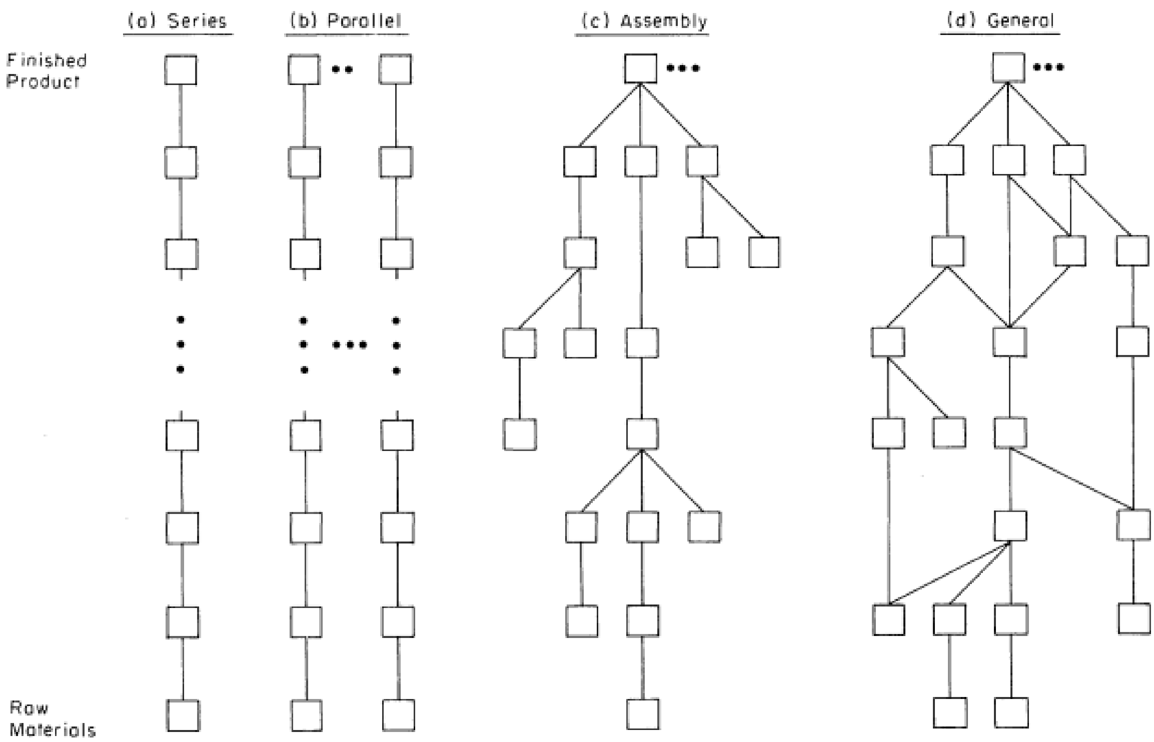


Figure 18: Four product structures described by Bellington et al. (1983, p. 1128)

Comelli et al. (2008) presents a Multi-Level Capacitated Lot Sizing Problem (MLCLSP) model formulation.

MSCLSP	Notation by Comelli et al. (2008)
Index sets	
N	Set of items $N = \{1, \dots, N\}$
T	Set of periods $T = \{1, \dots, T\}$
K	Set of resources $K = \{1, \dots, K\}$
S_i	Set of successor of item i in the BOM
Parameters	
d_{it}	External demand of item i in period t
b_{kt}	Available capacity of machine k in period t
z_i	Lead time of item i
v_{ik}	The capacity request for producing one unit of item i by machine k
a_{ij}	Production coefficient (number of units of item required to produce one unit of the immediate successor given by Gozinto matrix)
s_i	Non-negative setup costs for item i
h_i	Non-negative holding cost for item i
r_i	Non-negative production cost for item i
I_{i0}	Initial inventory for item i
M_{it}	The upperbound of the quantity for item i at period t
Variables	
X_{it}	Binary variable which indicates whether a setup for item i occurs at period t
Q_{it}	Production quantity of item i in period t
I_{it}	Inventory of item i at the end of period t

Model MLCLSP Notation by Comelli et al. (2008)

$$Min \sum_{t=1}^{T}\left[\sum_{i=1}^{N}(s_iX_{it} + h_iI_{it} + r_iQ_{it})\right] \quad (1)$$

Subject to

$$I_{it} = I_{i(t-1)} + Q_{i(t-z_i)} - d_{it} - \sum_{j\in S(i)} a_{ij}Q_{jt} \qquad (i,t) \in [1,N]\times[1,T] \quad (2)$$

$$\sum_{i=1}^{N} v_{ik}Q_{it} \le b_{kt} \qquad (k,t) \in [1,K]\times[1,T] \quad (3)$$

$$Q_{it} \le M_{it}X_{it} \qquad (i,t) \in [1,N]\times[1,T] \quad (4)$$

$$I_{it}, Q_{it} \in IN \qquad (i,t) \in [1,N]\times[1,T] \quad (5)$$

$$X_{it} \in \{0,1\} \qquad (i,t) \in [1,N]\times[1,T] \quad (6)$$

The objective function (1) is equal to the sum of the setup, holding and production costs that seek to be minimized. The first constraint (2) represents the inventory balances where internal demand is added and is modeled thanks to the Gozinto matrix. Here the product availability is modeled thanks to lead time. (3) represents capacity constraints. Constraint (4) represents the setup constraint: due to these restrictions, production of an item can only take place if the machine is set up for that particular item. Constraint (5) ensures non-negativity for the inventory and production quantity and (6) defines the setup variable as binary.

Buschkühl et al. (2008) also proposes a standard model formulation for the MLCLSP and uses production quantities and inventory levels variables making it a so called inventory and lot-size (I&L) formulation:

MSCLSP	Notation by Buschkühl et al. (2008)
Index sets	
K	Set of items $K = \{1,\dots,K\}$
M	Set of resource groups $M = \{1,\dots,M\}$
T	Set of periods $T = \{1,\dots,T\}$
K_m	Set of items k produced on resource m
S_k	Set of direct successors of item k
Parameters	
a_{kj}	Quantity of item k directly required to produce one unit of item j (Gozinto factor)
c_{mt}	Available capacity of resource m in period t
d_{kt}	External demand of item k in period t
h_k	Holding cost of item k per unit and period
s_k	Setup cost of item k
tp_k	Production time per unit of item k
ts_k	Setup time for the production of item k
z_k	Planned lead time of item k
b_{kt}	Sufficiently big number
Variables	
γ_{kt}	Binary setup variable of item k in period t
Q_{kt}	Production quantity of item k in period t
Y_{kt}	Inventory of item k at the end of period t

Model MLCLSP$_{I\&L}$ Notation by Buschkühl et al. (2008)

$$Min\ Z = \sum_{k \in K} \sum_{t \in T} (s_k \cdot \gamma_{kt} + h_k \cdot Y_{kt}) \tag{1}$$

Subject to

$$Y_{k,t-1} + Q_{k,t-z_k} - \sum_{j \in S_k} a_{kj} \cdot Q_{jt} - Y_{kt} = d_{kt} \qquad \forall k, t \tag{2}$$

$$\sum_{k \in K_m} tp_k \cdot Q_{kt} + ts_k \cdot \gamma_{kt} \leq c_{mt} \qquad \forall m, t \tag{3}$$

$$Q_{kt} \leq b_{kt} \cdot \gamma_{kt} \qquad \forall k, t \tag{4}$$

$$Y_{k0} = Y_{kT} = 0 \qquad \forall k \tag{5}$$

$$Q_{kt}, Y_{kt} \geq 0 \qquad \forall k, t \tag{6}$$

$$\gamma_{kt} \in \{0,1\} \qquad \forall k, t \tag{7}$$

In this model, the objective function (1) minimizes the total sum of setup cost and inventory holding cost. Constraint (2) is the inventory balance constraint which guarantees that the external demand d_{kt} and the secondary demands ($\sum_{j \in S_k} a_{kj} \cdot Q_{jt} - Y_{kt}$) of item k in every period t are met. Constraint (3) is the capacity constraint concerning the production and setup time for each resource m. Constraint (4) ensures that production of item k takes place in period t, only if the resource is setup for this item ($\gamma_{kt} = 1$). According to constraint (5) the initial inventory and the final inventory are assumed to be 0. Constraint (6) and (7) ensure that variables Q_{kt} and Y_{kt} are non-negative and the setup variable γ_{kt} is binary. Buschkühl mentions that further extensions to the above lot-sizing problem have been presented. Some authors account for overtime and backorder decisions ensuring a feasible solution in a mathematical sense as the corresponding decision variables have the function of slack variables. An example of such an extension is the original model presented by Bellington et al. (1983).

These two versions of the MLCLSP model have slight differences, but these are mainly in terms of formulation. The overall idea and structure of the models are very similar. Since these models cannot directly applied to a real-life situation. Both these models can form a basis for developing a model that is specific to a real-life situation such as the situation at Systems-2. The model of Buschkühl et al. (2008) has a notation that does not include costs related to the production which is similar to this project. But the way the constraints are formulated in the model of Comelli et al. (2008) seams to be more clear and easier to understand. For instance, both models use an auxiliary variable to calculate the inventory of an item in a certain period in constraint (2). However, this is more explicitly stated in the model of Comelli. And for instance both models take the quantity of an item into account when calculating the capacity in constraint (3), but the model by Comelli et al. (2008) is more similar to our situation as it is not a combination of setup times and production times but only one factor, just like we only need to consider the setup for our situation instead of a combination as in the model of Buschkühl et al. (2008).

Bruno et al. (2014) confirm that many variants of basic mathematical formulations have been developed but noticed that the structure of the models can be easily adapted even to fields not strictly related to inventory management, belonging to a wider logistic context. Through an appropriate interpretation of the elements of the model, Lot Sizing formulations can also be effectively used to face further practical logistic problems, outside the classical field of production and manufacturing. This can be done by adding constraints to the model in order to describe different production mode options. For instance by allowing a maximum number of setups per period or by limiting the total inventory level in each period. Another way of doing this is by redefining the variables. This way the basic model formulations can be extended such that the model becomes a more accurate representation of the real-life scenario.

Jaruphongsa et al. (2007) generalizes the classic dynamic lot-sizing model to consider the case where replenishment orders may be delivered by multiple shipment modes. For each mode, there is a fixed set-up cost associated with each delivery and a linear procurement cost for each unit supplied by that mode. This is the case if both replenishment modes have the "traditional" cost structure considered in the literature on the classical problem.

3.3 Solution approaches

The MLCSLP can be reduced to a smaller problem by setting some parameters to 0. According to Buschkühl et al (2008) this problem is NP-hard and the MLCLSP is at least as hard to solve and therefore also NP-hard. NP-hard problems cannot be solved to optimality in polynomial time and therefore heuristics are commonly used to provide solutions of reasonable good quality. A model and its solution approach are inherently linked: more complex models demand also more complex solution approaches to solve them (Jans & Degraeve, 2006). Buschkühl et al (2008) states that the approaches to solve different types of capacitated lot-sizing models can be classified into five groups: Mathematical Programming (MP) heuristics, Lagrangian Heuristics, Decomposition and Aggregation heuristics, Metaheuristics, and Problem-specific Greedy Heuristics. Two streams that appear to be particularly active are those that are based on mathematical programming and those that work with metaheuristics. These two approaches offer the flexibility to treat a broad variety of problems that arise in practice. They require and use the increased computing power that is nowadays available.

Jans & Degraeve (2007) review and compare six solution approaches: Meta-Heuristics, Dynamic programming, Polyhedral results and strong valid inequalities, Lagrange relaxation and Dantzig-Wolfe decomposition, Reformulations of the lot-sizing problem, Special-purpose heuristics for lot sizing problems. For the standard problems, traditional method or special purpose heuristics seem to outperform meta-heuristics. On the other hand, metaheuristics provide good results for the multi-level or sequence dependent problems, for which the traditional methods fail. Hybrid systems have been developed to combine the strengths of different methodologies. Solutions obtained by LP-based heuristics provide good starting solutions for meta-heuristics.

MIP-based heuristics for the MLCSLP solve a series of subproblems (submodels) whose number of binary variables is much less than that of the original MIP model of the problem. Among these heuristics, relax-and-fix approaches and fix-and-optimize (FO) approaches are mostly used (Chen H. , 2015). In a relax-and-fix heuristic, the binary setup variables of the original MLCLSP are divided into three groups for each subproblem: the first group contains those that are fixed, the second those that are optimized and the third contains those for which the integrality constraints are relaxed. This leads to an underestimated capacity consumption due to setup times. For this reason a reformulations of the MLCLPS are used to strengthen the lower bounds. This is not needed for the FO heuristic as it operates only with the first two groups. The major advantage of the FO heuristic is, hence, that in each iteration a formally feasible solution is obtained (Helber & Sahling, 2010).

The basic idea of the FO heuristic proposed by Helber & Sahling (2010) is to solve in an iterative fashion a series of subproblems that are derived from the MLCLSP. Their MLCLSP formulation is similar to the notation by Buschkühl et al. (2008) as it is both based on the model introduced by Bellington et al. (1983) but account for overtime to ensure feasibility. In each iteration, most of the binary setup variables γ_{kt} are set to a fixed value $\bar{\gamma}_{kt}^{fix}$. This reduces the number of "free" binary variables in the subproblem of the current iteration. As the number of "free" binary variables of the subproblem is much smaller than the original MLCLSP, the solution time for a subproblem is very small. This yields a new temporary solution for the setup variables of the current subproblem. The bigger the relative number of free variables in each subproblem MLCLSP-SUB is, the more time consuming is the solution of the resulting MIP and the higher is the quality of the solution that can be found. Helber & Sahling (2010) experimented with several different ways to define subsets of binary variables to optimize out of which three turned out to be particularly useful: Product-oriented decomposition, Resource-oriented decomposition, process-oriented decomposition. Each of these decompositions reflects a particular perspective on the problem. Chen H. (2015) also proposes a FO approach for the MLCLSP, based on the model introduced by Bellington et al. (1983) including overtime, different from the one of Helber & Sahling. This FO approach selects the binary variables to be re-optimized in an MIP model of a lot-sizing problem based on the interrelatedness of binary setup variables in the constraints of the model rather than based on three problem-specific decompositions. This approach is thus more general and can be applied to other 0-1 MIP models. Numerical experiments on benchmark instances show that the FO approach of Chen H. (2015) can obtain a better solution in a similar computation time for most instances compared with that found by the approach of Helber and Sahling. Moreover, Chen H. also developed a variable neighborhood search approach for the MLCLSP, which can further improve the solution obtained by the FO by diversifying the search space making it a hybrid system.

Fix and optimize algorithm for the MLCLSP by Chen H. (2015)
In the algorithm, the number of possible subproblems is $N \times T$, where N is the number of items and T is the number of periods. Initially, a setup is planned for each product in each period, i.e., we set $Y_{it} = 1$ for all i and t. In each iteration, a setup variable Y_{it} or a pair (i, t) is randomly selected from $N \times T$ with the same probability for each element in $N \times T$, and the corresponding subproblem $SP^l_{i,t}$ is solved, where the level of interrelatedness l is a control parameter of the algorithm. For more information on the interrelatedness and the definition of subproblems, see Chen H. (2015). The subproblem can then be inserted into the MIP solver.

The subproblems solution is accepted if and only if it is better than the current best solution of the original problem. Here, the first solution is better than the second solution if one of the two conditions holds: (1) the first solution is capacity-feasible (i.e., without overtime), and its cost is lower than the cost of the second solution; (2) the first solution yields a cost lower than that of the second solution if they are both not capacity-feasible. In other words, a capacity-infeasible solution is never considered a candidate for the best solution if there is already a known capacity-feasible solution. The algorithm will terminate if no improvement is observed after a certain number of iterations. This number, denoted by n, is another control parameter of the algorithm and is dependent on the number of possible subproblems.

Let $\bar{Y} = \{\bar{Y}_{it}\}$ denote the values of all setup variables from a solution of model MLCLSP. $\bar{Y}$ is also called a setup plan or solution of the model that includes the value of every setup variable. Let $iter$ denote the number of iterations performed since the last improvement. Note that in this approach overtime is included in the MLCLSP model. The addition of overtime allows the model to find a solution even though it is not capacity feasible. If there is overtime in the objective function of the model, we know that this is not a capacity feasible solution. If there is no overtime, we know it is a capacity feasible solution. The FO algorithm is presented in pseudo-code in Algorithm 1.

Algorithm 1. FO-Algorithm (l, n)

Find an initial feasible solution of model MLCLSP by setting $Y_{it} = 1$ for all i and t;
Set the current best solution $\bar{Y} = \{\bar{Y}_{it}\}$ of the model to the feasible solution;
$iter \leftarrow 0$;
Repeat
    $iter \leftarrow iter + 1$;
    Randomly choose a pair (i, t) from $N \times T$ with the same probability for each element in $N \times T$;
    Solve subproblem $SP^l_{i,t}$;
    If *the solution of $SP^l_{i,t}$ is better than the current best solution $\bar{Y}$ of model MLCLSP,* ***Then***
        Set $\bar{Y}$ to the solution $SP^l_{i,t}$;

$iter \leftarrow 0$;
End If
Until $iter \geq n$;

Variable neighborhood search (VNS) for the MLCLSP by Chen H. (2015)
Despite its relatively large neighborhood structure, the FO based local search for the MLCLSP can only find a local optimum of the problem in most cases. In order to find a global optimum or a solution close to the global optimum, we must diversify the search space so that more promising regions can be explored. As most VNS approaches, this VNS for the MLCLSP pre-selects a finite set of neighborhood structures $N_k, k = 1,2, \dots, k_{max}$ where k_{max} is the number of neighborhood structures considered and $N_k(\bar{Y})$ denotes the set of neighboring solutions in the kth neighborhood of a solution $\bar{Y}$ of the problem. For each neighborhood structure given, a local optimum is found by a local search starting from the current solution. The diversification is realized by applying a shaking algorithm that creates a new starting solution for the local search by randomly perturbing the current solution. To describe the VNS approach let's define:

$\bar{Y}$	Setup plan of the current solution,
$\bar{Y}^*$	Setup plan of the incumbent (the current best solution),
CT	Computation time so far,
CT_{max}	Maximum computation time allowed,
k_{max}	Number of neighborhood structures considered,
k	Index of neighborhood structure N_k

With the above notations, our VNS approach for the MLCLSP is presented in pseudo-code in Algorithm 2. In this algorithm two algorithms (algorithm 3 and 4) must be executed. A brief explanation on how these algorithms work is given, the full pseudo-code of these algorithms can be found in Appendix A.

Algorithm 2. Variable neighborhood search approach for the MLCLSP

Find an initial feasible solution $\bar{Y}_0$;
Set $\bar{Y} \leftarrow \bar{Y}_0$, $\bar{Y}^* \leftarrow \bar{Y}_0$, *and* $k \leftarrow 1$;
Repeat
(local search) Apply a fix-and-optimize local search algorithm (Algorithm 3) with $\bar{Y}$ *as the initial solution to obtain a local optimum* $\bar{Y}'$ *of model MLCLSP with the neighborhood structure* N_k
If $\bar{Y}'$ *is better than the incumbent* $\bar{Y}^*$, ***Then***
Set $\bar{Y}^* \leftarrow \bar{Y}'$, $\bar{Y} \leftarrow \bar{Y}'$, *and* $k \leftarrow 1$;
$iter \leftarrow 0$;
Else
Set $\bar{Y} \leftarrow \bar{Y}^*$ *and* $k \leftarrow k + 1$;
If $k > k_{max}$, *then* $k \leftarrow 1$;
End If
End If
(shaking) Generate a new starting solution $\bar{Y}''$ *from the current solution* $\bar{Y}$ *by a random swap-fix-and-optimize routine (Algorithm 4) and set* $\bar{Y} \leftarrow \bar{Y}''$;
Determine the current computation time CT;
Until $CT \geq CT_{max}$;

Local search by fix-and-optimize
Since the local search algorithm is called many times in the VNS, a direct adoption of the FO algorithm may be too time consuming. The computation time of each local search loop can be

reduced by limiting the number of subproblems to be solved in each iteration of the fix-and-optimize algorithm.

Shaking by random swap-fix-and-optimize
In this VNS approach, the starting solution $\bar{Y}''$ for each local search loop is generated by a shaking routine that performs a series of random swap-fix-and-optimize operations on the current solution $\bar{Y}$. Here the swap of a setup variable Y_{it} means that its value is changed from $Y_{it} = \bar{Y}_{it}$ to $Y_{it} = 1 - \bar{Y}_{it}$, where $\bar{Y}_{it} \in \{0,1\}$ is the value of Y_{it} at the current solution. Swapping a setup variable also causes the model to optimize the production variable X_{it}, as an item can only be produced when there is a setup in for that item in a specific period. So the solution does not only alternate in the setup variables but also in the variables that are related to the setup variables. Since we want to find a high quality capacity-feasible solution of the model MLCLSP, a perturbed solution is acceptable only if one of the following two conditions is satisfied: (1) the current solution is capacity-infeasible, (2) both the current solution and the perturbed solution are capacity feasible.

3.4 Stochasticity

The demand quantities in this research are known and thus deterministic. But the timing of the number of materials is not known beforehand, meaning that we are dealing with stochasticity, or dynamic demand. Neglecting uncertainty in input parameters leads to suboptimal production plans that perform poorly when they are evaluated in situations where the parameters actually vary from their mean. Stochastic optimization takes into account the stochasticity by sampling discrete scenarios from the underlying probability distribution and incorporating a metric over these scenarios, such as the expected value, into the optimization model (Schlenkrich & Parragh, 2024). A way to model this uncertainty is by extending the model to a two-stage programming model. In the application of tactical models, the first stage variables define the baseline production plan, i.e., schedule of production and material purchase, and the second stage variables represent possible updates of production after the realization of uncertainties (Hu & Hu, 2016). In practice it is common to use scenarios to represent these uncertainties because a parameter with continuous distribution is really hard to be applied from both modeling and computational perspective (Escudero and Kamesam, Cited in Hu & Hu, 2016).

According to Emelogu et al. (2016), Sample Average Approximation (SAA) is a popular approach which is frequently employed to solve large scale stochastic optimization problems. In this technique, the expected objective function of the stochastic problem is approximated by a sample average estimate derived from a random sample. The resulting sample average approximating problem is then solved by deterministic optimization techniques (Verweij et al.,2003). A disadvantage of the SAA method is selecting the right sample size. Emelogu et al. (2016) states that choosing the appropriate sample size in SAA method is very challenging. An inappropriate sample size can lead to the generation of low quality solutions with high computational burden.

3.5 Relevant performance indicators

By using an FO approach in combination with VNS we aim to find a good or close to optimal feasible solution for the MLCLSP. It is therefore important to measure the performance of the solution approach. The choice of performance measures for experiments on heuristics necessarily involves both solution quality and computation time (Rardin & Uzsoy, 2001). Almost every computational experiment compares performance to the best known solution for each test instance. Sometimes, just like in the papers of Helber & Sahling (2010) and Chen H. (2015), the best known solution comes from other researchers. However, since we are dealing with a real-life situation, the use of long runs can be a more suitable source for best known solutions. A local search might be applied many times from different starts to obtain a good approximation to the optimal value, or a continued method like Tabu Search might be run long beyond the stopping point. Very large computation times can be justified if they are needed only once on each instance (Rardin & Uzsoy, 2001).

In the proposed solution approach we have the possibility to influence the computation time spent by modifying the number of iterations or the number of neighborhood structures considered and ultimately the computation time. Since we are applying this solution approach to a real life scenario, the tradeoff between time and solution quality becomes relevant. This trade-off can be illustrated by a simple graphic displaying the time or objective function values on the horizontal axis and the solution value on the vertical axis. A plot line tracks the sequence of objective function values visited as the search proceeds (Rardin & Uzsoy, 2001).

The model for this research must represent a real-life situation. For many cases a real life scenario includes many factors that can add difficulties. Therefore, we must simplify the problem and make certain assumptions in order to create a mathematical model. All mathematical models are approximate and their usefulness depends on our understanding of the uncertainty inherent in the predections (Arriola & Hyman, 2009). We must find out to what extend our model accurately represents the reality. One way to do this is to make use of Sensitivity Analsysis (SA). SA can be used to quantity the effects of uncertainties on a model's input parameters and the subseqent effect on the model's output. That is, SA can determine how variability of the inputs causes variability in the outputs.

3.6 Conclusion

Table 3 summarizes the literature that is used during this literature search. The structure of the table represents the structure of this literature search. First, general reviews about the lot-sizing literature are consulted to define our lot-sizing problem. Second, literature about modifications to lot-sizing problems is consulted. At last, we have searched for literature that describes possible solution approaches that can be applied to our model. In this chapter we have answered the second research question: *What methods are proposed in the literature to control WIP inventory at the production area?* This project will be modeled as a Multi-Level Capacitated Lot-Sizing Problem (MLCLSP) to determine the correct number and timing of delivering materials to the production area. The presented MLCLSP models cannot directly be applied to the situation at Systems-2 but must be combined, reformulated and possibly modified by changing certain variables and constraints, but serve as a base for developing the model specific to this project. Also scenarios will be included to deal with uncertain timing of demand. This will be done in Chapter 4. Most comparable studies use a general version of the MLCLSP model and provide methods such as metaheuristics in order to find good solutions. These studies mainly focus on showing that a certain solution approach finds good solutions by applying it to predefined test-sets. These studies have all tested their models on test problems appropriately generated to represent practical case studies or in order to show that the model works or that the models performance was improved. In this project, the MLCLSP model will be modified such that it is an accurate representation of the problem at Systems-2. Due to the MLCLSP being NP-hard, a Fix-and-Optimize approach could be used as inspiration for a solution approach. In this process a tradeoff must be made between the quality of the solution and the computation time.

Once a solution, or setup-policy is found, we must find out how it performs and to what extend the model provides consistent results. This validation will be done by applying a sensitivity analysis. This can be usefull as it allows us to modify parameters such as the available capacity and the planning horizon which could be interesting as it can serve as input for making decisions on reducing or improving capacity and because the througput time (planning horizon) of a module might change in the upcoming years.

Table 3: Overview of literature used during this literature search

Article	Goal of the research	Similarity to this research	Modeling approach	Solution approach	Applied to
Karimi et al. (2003)	Reviewing of LSP's together with exact and heuristic approaches.	Tackling the lot sizing problem	Single-level lot sizing problems.	Exact methods, common sense, MP heuristics.	N/A
Jans & Degraeve (2006)	Reviewing the modeling of LSP's.	Tackling the lot sizing problem	LSP, CLSP, and extensions.	N/A	N/A
Comelli et al. (2008)	Create an overview to solve tactical planning problems.	Tackling the lot sizing problem	MLCLSP according to Tempelmeier & Derstroff (1996)	Common sense, MP, and meta heuristics.	N/A
Büschkuhl et al. (2008)	Reviewing different modeling and algorithmic solution approaches.	Tackling the lot sizing problem	MLCLSP according to Bellington et al. (1983)	MP, Lagrangian, decomposition and aggregation, met, and greedy heuristics.	N/A
Bruno et al. (2014)	Adaptations of CLSP for other logistical applications.	Adapting the CLSP to real-life problems by modifying or adding variables and constraints.	CLSP extended to Bus terminal schedule, cross-docking operations, and check in service.	Exact methods specific to the problem.	Test problems appropriately generated to represent practical case studies.
Jaruphongsa et al. (2007)	Consider multiple shipment modes for replenishment of orders.	Including multi-mode replenishments.	Classical dynamic lot sizing problem.	Decomposition and dynamic programming.	Problem specific test cases.
Helber & Sahling (2010)	Presenting an optimization-based solution approach for the MLCLSP.	Based on the same MLCLSP notation. Makes use of FO approach	MLCLSP according to Bellington et al. (1983)	Fix-and-Optimize approach.	5 test sets in order to evaluate the performance.
Chen H. (2015)	Solving the MLCLSP by a new fix-and-optimize approach.	Based on same MLCLSP notation. Makes use of FO + VNS approach	MLCLSP according to Bellington et al. (1983)	Fix-and-Optimize approach, FO + VNS.	5 test sets in order to evaluate the performance.
This research	Reducing WIP inventory at a production, cleaning and unpacking area.		Bellington et al. with modifications in sets, parameters, variables and constraints.	Fix-and-optimize + VNS and including multiple demand scenarios	A real-life scenario with multiple replenishment modes and with uncertain demand.

4 Model design

In this chapter, the design of the model will take place to provide an answer for the third main research question:

> *What should the model for controlling the WIP inventory at the production area while improving the flow of materials look like?*

Section 4.1 describes how the situation at Systems-2 is interpreted as a dynamic multi-level capacitated lot-sizing problem. Section 4.2 describes the assumptions that are made. Section 4.3 describes how the input data for the model is prepared. This includes establishing the Gozinto matrix. Section 4.4 focuses on the notation of the MLCLSP model specific to Systems-2. Here, the sets, parameters and variables will be discussed. Section 4.5 shows the solution approach for the MLCLSP model that is used in our research.

4.1 The situation at Systems-2 translated to an MLCLSP model

The key aspects of the multi-level capacitated lot sizing problem are the setup, the capacity and the inventory. This paragraph will elaborate on how these three key aspects are present in the situation at Systems-2, and how these are translated to the model.

Setup

In most industries the setup costs and times are related to the production area, due to having to make changes to machines or environments to produce a certain product. At Systems-2 the setup is different. The production environment does not have to change, since they are dedicated to a specific module. Instead, at Systems-2 all the materials that enter the production area must be unpacked or cleaned in the unpacking and cleaning area which is done by one FTE. This requires time and therefore incur cost. The time required for unpacking and cleaning are modelled as setup times and the cost of the time spent for the unpacking and cleaning are modelled as setup cost in the MLCLSP.

Capacity

The capacity is related to the unpacking and cleaning process and not related to the activities that take place in the production area. In this project we focus on controlling the WIP inventory at the production area but also the unpacking areas. The current distribution of materials over the processing steps causes high peaks of workload at the unpacking and cleaning area. These areas have limited space and are designed to execute the process of unpacking and cleaning and not to store materials. If too many materials need to be unpacked or cleaned at once, inventory will build up in the unpacking and cleaning area. Our model includes a setup time per material. To prevent WIP inventory in the unpacking and cleaning area, we allow a maximum time spent for the unpacking and cleaning of materials.

In more general lot-sizing models, the capacity is related to the activities related to the production process, such as the maximum time a machine can operate in a given time. As shown in the lay-out, every module has its own dedicated space where it is assembled. There is no need for switching to a specific work place and thus no maximum capacity for the time spent at a workplace for a module.

Inventory

The materials that go to the production area do not leave the production area until the module is assembled completely and shipped to the customer. So in principle, the physical inventory does not decrease until the module leaves the production area. However, in reality if an item is assembled, the item is used and has added its value to the module. So, in this project, assembling an item to a module lowers the inventory. The BOM of the modules at Systems-2 follow an general product structure as shown in the example in Figure 19.

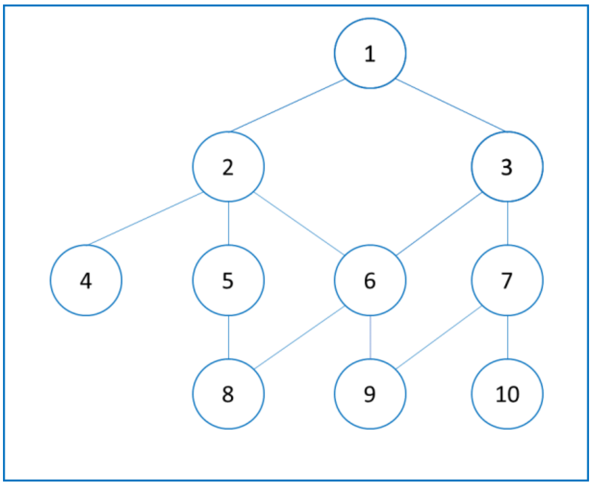


Figure 19: Simplified example of the general product structure

If we look at this simplified version of the product structure. We can categorize three types of items: The end-item, the assembly items, and the items with no predecessors, which we will refer to as 'Purchase items'.

The end item is the module that is completely assembled and will be delivered to the customer. Among the assemblies we can distinguish two types of assemblies. Sub-assemblies that consist of multiple purchase items but is not a newly created item. So for instance item 8 and item 9 are needed in combination with item 6 but after combining them it does not create a new item. There are also items that are created in the production area. Before production starts, this item does not exist. An example of such an item can be item 2, which is created after combining item 4, 5, and 6. At Systems-2 these items are called: Phantom items. This distinguishment is of high importance for this project. This is because the setups and capacity in this model are related to the unpacking and cleaning process of physical items. These phantom items are created in the production area and are not physically transported into the production area and therefore do not require a setup and thus do not influence the capacity of the unpacking area. But when such a phantom item is created, its inventory does go up and the inventory of its predecessors decreases.

4.2 Assumptions

We have now globally discussed how the situation at Systems-2 is translated to the MLCLSP. But there are still some assumptions that must be made in order to simplify the situation to make the model applicable.

Modules

At systems-2 there are logistically 5 modules that could potentially benefit from implementing this model. However, due to a limited available time for this project, our model is not implemented for every module. Instead we selected 2 modules for the model to be implemented on. For this project and Systems-2 it is interesting to get insight in how the results of the model could impact the production of a module and how well the model performs.

To get an idea of how well the model performs we apply this model to Module 1. This is a relative large module in terms of size, value and number of materials and also has a relatively high throughput time. This module uses a combination of purchase items and assembly items, has items that are transported through both the unpacking area and the large and medium cleaning area, and encompasses items that are transported through crates, pallets and transport pallets for piping making it a good example for how well the model performs as it encompasses all the relevant elements of this model.

In addition to Module 1, this model is be applied to Module 4. This is the smallest module in terms of size, value, number of materials, and throughput time. Due to the smaller number of materials in this module, this module can be used as a smaller test instance for the numerical analysis performed in Chapter 5.

Setup time for unpacking of crate materials
Internal calculations showed that the required time for a single material to unpack one layer is two minutes including additional tasks. The full unpacking process consists of two unpacking moments, but at the second unpacking moment the additional tasks are not required. This makes the second unpacking moment slightly faster than 2 minutes. The factory engineers at Systems-2 have therefore decided to use 3.4 minutes instead of 4 minutes as estimation for the full unpacking process of a single material that is transported through the crates.

Setup time for unpacking of pallets
Besides the materials that are delivered in crates, there are also materials that are delivered on a pallet due to their size. These are the materials that do not fit in a crate but do not require additional facilities to handle the material. Experience of the workers in the unpacking area learns that it takes approximately 5 minutes for the full unpacking process of a single item. This is also the value that we use for this model.

Setup time for unpacking of transport pallets for piping
The piping that is required in the modules are not delivered separately but are aggregated and delivered on a tailor made transport pallet which is packed in several layers to comply with the cleanliness restrictions. Due to this aggregation, we can assume the time necessary for the unpacking in the following ways. We assume the time necessary for the unpacking of this tailor made transport pallet is equal to that of a normal pallet and:

- Divide this by the number of pipes that is on the pallet.
- Or we aggregate the pipes in the data preparation for the model as well such that such a pallet becomes one item.

The first assumption brings the problem that the model could split the aggregated crate by selecting pipes in different periods which could cause practical problems when implementing the results of the model. But the second assumption causes additional complexity in the data preparation that makes the model prone to mistakes when preparing the data when the model is extended to other modules by an employee of Systems-2. Therefore we use the first assumption, as we are more interested in how we can lower the WIP at the production area in general.

Setup time for unpacking / cleaning special
Items that do not follow the standard process through the unpacking area due to their size or weight must enter the production are through the large & medium cleaning area. Here the necessary facilities are available to handle these special items. Based on the experience of the workers, it has become clear that handling a single item in this area requires two people and takes approximately 15 minutes. We do have to consider that this is a two person job so the setup costs will be higher.

Capacity
For the capacity that is available in the unpacking area and the large & medium cleaning area, we are interested in the capacity that is available for a single module. Currently there is 1 FTE available, or 40 hours, for the unpacking and cleaning of materials that need to be delivered to the production area. For this model we assume that we can use 100% of this time for the activities in the unpacking area and L&M Cleaning area. In reality this percentage will be lower due to productivity and other unforeseen causes. In consultation with the Logistic Engineer we have made the decision that a

80/20 split for the activities is realistic. This gives us the following available capacity for a period at the Unpacking area and L&M Cleaning:

$$Unpacking\ area: 0.8 * \frac{40}{5} = 6.4\ hours\ or\ 384\ minutes$$

$$L\&M\ Cleaning: 0.2 * \frac{40}{5} = 1.6\ hours\ or\ 96\ minutes$$

Notice that we divide by five, this is because we look at the available time per module. Logistically speaking, there are 5 modules. So, we have 384 minutes available per period for a single module in the unpacking area, and 96 minutes available per period for a single module in the L&M Cleaning.

Periods

In the introduction of this project it is mentioned that the main motivation for this project is the fact that Systems-2 aims to reduce the Throughput Time (TT) over time. The goal is to assembly modules according to a move-rate of 1.2 in 2025 and a move-rate of 1.67 in late 2026. The move-rate basically determines the TT of a module as it determines how many modules must be produced in a year, and thus the maximum time it can take for producing a single module. However, the move-rate is dependent on other factors such as the number of ovens, possible additional production space and maybe extra mechanics. For this project we have a finite planning horizon, meaning that we must make an assumption for the number of periods that are included in the model. Therefore choosing a number of periods based on a move-rate that is not yet realized is not ideal as the results could possibly not be implemented due to these other factors. Therefore, in this project we set the number of periods equal to the current TT of a module. Which for module 1 is equal to 18 periods, and for module 4 is equal to 12 periods.

Uncertain timing of demand

In this project the demand quantity of the materials are deterministic. But, due to uncertainties in the material and production planning process described in Section 2.1 it is very hard to tell at which exact moment in time a certain material is needed. So the time at which these materials need to be brought to the production area is non-deterministic. To represent this uncertainty, scenarios are included in our model. In Chapter 2, we describe that the actual starting moments of the processing steps can vary from 2 to 7 days. This has to do with the fact that there are processing steps that every module must undergo, but only one module can undergo at a time. If that processing step is occupied, the production of the module is paused resulting in a delay. And if that processing step is free while it is scheduled to be occupied, the production of the module progresses earlier than planned. Therefore we include three scenarios: a scenario where a processing step starts a period (one week) earlier, a scenario where a processing step starts in the scheduled period, and a scenario where a processing step starts one period later. Including scenarios for every single item will make the problem extremely large. Therefore we only include these scenarios for the phantom items. A phantom item is a result of combining several items (predecessors in the BOM) together and does not physically exist until it is produced. These phantom items are present in lower levels of the BOM, meaning that they are directly or almost directly needed for producing the end item. Adding scenarios to the phantom items influences all of its predecessors by causing a chain reaction. If a phantom item is needed at an earlier point in time, its predecessor is also needed earlier, and the predecessor of this predecessor also, and so on. This way, we prevent the model to grow extremely large but do model the uncertainty in the timing of the demand.

Setup cost

The setup costs for this project are related to the time spent for the setups that are previously discussed. The cost per hour spent on unpacking materials equals €86,50. So it would cost approximately €1,44 per minute spent on unpacking. The unpacking of a material in a crate would cost €1,44 * 3.4 = €4,90 in this model. In the upcoming parts of this report it will become clear that the total quantity of a material is already given in the data at systems-2. So, if the BOM says that a quantity of 40 is needed for a parent, and a quantity of 10 is needed from the parent, then these numbers do not need to be multiplicated as normally done in the models described in the literature review. Thus, the total quantity of an item is already in the BOM.

For the setup cost this is an important aspect, as the quantity of materials that are unpacked or cleaned depending on their load carrier determine the total setup cost of an item. So, the quantity of an item that is needed for a parent is fixed. But since we have a general product structure, it occurs that one item has multiple parent. Just like item 6 has item 2 and 3 as its parent in the simplified example of Figure 19 and the quantity that is needed for item 2 can differ from item 3, and thus the setup cost for item 6 depends on its parent. A logical way of modeling this dependency is by including the order quantity in the objective function and multiply this with the setup cost for a single item. However, a decision has been made to implement a Fix-and-Optimize approach where we first fix the setup variables and later select setup variables to optimize. These setup variables are binary. So the setup costs are fixed in advance. Here we can multiply the setup cost with three possible quantities:

1. We take the lowest quantity that is needed of an item for any of its parents;
2. We take the average quantity that is needed of an item for any of its parents;
3. We take the highest quantity that is needed of an item for any of its parents.

In this research we make the assumption to take the highest quantity that is needed of an item for any of its parents and multiply this with the setup cost per quantity of this item. This way, we cover the worst case scenario. Our model will use higher setup cost for some of the items, but the total cost in the objective value cannot be higher in reality. Which could be the case in the first, and second option.

Holding cost

At VDL ETGA the Weighted Average Cost of Capital (WACC) is used to determine the cost of inventory. This includes the time an item spends at VDL ETGA. Since we are using periods in our model and we are only interested in the time an item spends as inventory at the production area before being assembled, we use the percentage related to the actual costs of holding an item in inventory. This is 8%.

4.3 Data preparation

For this project we use a MLCLSP model to model the material and production planning of Systems-2. By using this model we can find the optimal quantity and timing of materials to deliver to the production area for every module separately. But before this model can be used some adjustments to the input data must be made in order to accurately represent the situation at Systems-2. Section 4.3.1 elaborates on how the Gozinto matrix is constructed for a single module. Section 4.3.2 elaborates on the phantom items and their due dates. Section 4.3.3 concludes the data preparation with an elaboration on how the holding and setup costs are determined.

4.3.1 Gozinto matrix

As already mentioned, every module at that is produced by Systems-2 has a Bill-of-Material with a general product structure. This product structure is build out of parent-component relationships with up to 12 levels. For the model it is important that we get an overview of which item is directly required for the production of another item. As described in the literature review, the BOM can be algebraically translated to a so called Gozinto matrix. However before this can be done, it is important to filter and select the right data as input for the matrix. The BOM of a module is stored in the ERP-System BAAN. In BAAN it is possible to export the BOM to Excel which is necessary for filtering the data. For this module we are only interested in the items that are module specific and need to enter the production area through one of the unpacking areas. The items that are in the BOM are categorized in BAAN by giving them a warehouse location number such as 625. For this project we are only interested in the items that are assigned to the following warehouse locations:

605: Purchase items
625: Phantom items

Now that we have filtered and selected the items that are included in this project. A few additional items must be excluded. These are articles that:

- Have a value of 0,
- Have a unit of grams or liters,
- Are used outside the cleanroom (such as carton edge pieces used for packing the module before transport).

Now the remaining items can be exported to another sheet including their parent, item-id, and their required value. In this problem we deal with over a 100 different items and if all parent-component relations are transformed into matrix, there will be many '0' values which can already be seen in the matrix with 6 items of Figure 17. Therefore three columns are created with the Item-id, Parent-id and the required quantity of the item id that is needed for the parent-id. Figure 20 illustrates how the Gozinto matrix presented in the literature review will be translated into a table form.

$$
\begin{array}{c}
\\ \text{E. I.} \\ \text{A} \\ \text{B} \\ \text{C} \\ \text{D} \\ \text{E}
\end{array}
\begin{array}{c}
\begin{array}{cccccc} \text{E. I.} & \text{A} & \text{B} & \text{C} & \text{D} & \text{E} \end{array} \\
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 0 \\
2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 4 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 0
\end{pmatrix}
\end{array}
$$

Gozinto Matrix

Item	Parent	Value
A	E. I.	3
B	E. I.	2
C	B	4
D	A	1
E	A	2

Figure 20: Matrix transformed to a table

4.3.2 Phantom items and scenarios

One of the item categories that are in the BOM is the Assembly-category. In this category, a distinction can be made by regular assemblies and the so called phantom items that are described in Section 4.1. As said, these phantom items are items that are produced in the production area by assembling the required predecessors and do not physically exist before they are produced. The sequence in which items must be assembled and thus delivered to the production area is relevant. The phantom-items of the modules at Systems-2 are typically located high up in the Bill-of-Material. If we look at the simplified example of the general product structure in Figure 19, we can consider item 2 and 3 as phantom items. If we only use a due date for end item, item 1 in this case, the phantom items will probably be assembled and ready at the same time. But let's say item 2 is a rather simple item, and item 3 is a complex item that is crucial for the quality of the end item and thus must undergo extra processing steps such as a quality inspection and a measurement for certain specifications. Then this item must be produced at an earlier moment in the production process meaning that items 6, 7, 9, and 10 must also be delivered to the production area at an earlier point in time. This, again, simple example is representative for items in the production process at Systems-2. Including uncertainty by adding scenarios for the phantom items influences the second stage variables related to predecessors of these phantom items, while keeping the number of scenarios limited.

4.3.3 Holding costs, setup cost and setup time

As mentioned before, setup and holding costs are included in this project. We cannot just use the parameters that are mentioned in the assumptions yet. For every item we must assign the right unpacking time, we must then multiply this with the hourly rate that is mentioned in the assumptions to find the setup cost of an item. The holding cost is a percentage, 8% in this case. This means that the holding cost of an item is dependent on the value of that item. Therefore, similar to the setup cost, we must first multiply the holding cost percentage with the value of an item to get the holding cost for a single item.

4.4 MLCLSP Model

In this paragraph we present the finalized MLCLSP model. First the necessary additions and modifications for the model are described in4.4.1. After discussing the additions and modifications we first present the sets, parameters and variables of the model. Followed by the objective and constraints that complete the model.

4.4.1 Additions and modifications

The two versions of the MLCLSP model in the literature have provided a good starting point for the development of the model. However a few additions had to me made to make the model specific for this project. First a set of scenarios S has been added for modeling the uncertainty in the timing of demand. Also a subset of parent items A have been added. Then, we modified the set of resources or machines by specifying the use of unpacking areas in set M.

In the parameters, we add an overtime cost such that the model can find a feasible solution. This overtime cost oc_m can be interpreted as a penalty cost for neglecting the capacity constraint. We also added a parameter for the production time of a phantom item indicated by p_{is}. This parameter must be added to be able to model scenarios for the production of phantom items. At last we have added the auxiliary variable for the overtime at an unpacking area m in a certain period. This variable works in combination with the overtime cost in the objective function that will be discussed in the following paragraph.

In the balance constraints of the two MLCLSP versions that are described in the literature review the Gozinto factor is multiplied by the number or parent items that are produced. The BOM data at Systems-2 gives the total quantity of an item that is required in combination with the parent. So instead of having a multiplication with the total quantity of the parent item, we must multiply the quantity of the item by 1 if that item is triggered by the parent item. Therefore we introduce a new binary variable X_{jt} that is 1 if a parent item j is ordered in a period t and 0 otherwise.

4.4.2 Sets, parameters and variables of the MLCLSP

During the literature search two versions of the MLCLSP model were presented. Between these two versions there are a few distinguishments related to the sets, parameters and variables. For the development of the model we took components from both versions or combined components from the two versions. Besides that we still needed to add certain sets, parameters, and variables to make the model applicable to the situation at Sytems-2.

MSCLSP Systems-2

Sets

N	Set of items $N = \{1, \ldots, N\}$
A	Set of parents $A = \{1, \ldots, A\}$
T	Set of periods $T = \{1, \ldots, T\}$

M	Set of unpacking areas $M = \{Unpacking\ area, L\&M\ Cleaning\}$
S	Set of scenarios $S = \{1, \dots, 3\}$
Indices	
i	Indice indicating item i in set N
j	Indice indicating parent j in set A
t	Indice indicating period t in set T
m	Indice indicating unpacking area m in set M
s	Indice indicating the scenario s in set S
Parameters	
a_{ij}	Quantity of item i directly required to produce one unit of item j (Gozinto factor)
c_{mt}	Available capacity of unpacking area m in period t
d_{it}	Demand for end item i in period t
h_i	Holding cost of item i per unit
sc_i	Setup cost of item i
st_{im}	Setup time for the unpacking of item i in unpacking area m
p_{is}	Production time of item i in scenario s
oc_m	Overtime cost of unpacking area m
B	Sufficiently big number
Variables	
y_{it}	Binary setup variable of item i in period t
X_{it}	Auxiliary binary variable for item i in period t
Q_{it}	Order quantity of item i in period t
I_{its}	Inventory of item i at the end of period t in scenario s
O_{mt}	Overtime of unpacking area m in period t

4.4.3 Objective and constraints

The objective for this project is to find the optimal policy at which the right materials are delivered to the production area at the period in time and in the right quantity by minimizing the setup, holding, and potential overtime cost such that the end-item can be produced in time. We speak of a policy instead since it considers multiple scenarios while finding a solution to deal with dynamic demand. This led to the following objective function for the MLCLSP model:

$$Min\ Z = \sum_{i \in N} \sum_{t \in T} (y_{it} \cdot sc_i) + + \sum_{m \in M} \sum_{t \in T} (oc_m \cdot O_{mt}) + \sum_{i \in N} \sum_{t \in T} \sum_{s \in S} (\frac{1}{3} I_{its} \cdot h_i) \qquad (1)$$

This objective function is subject to the following constraints. Constraint (2) is the inventory balance constraint. The inventory of an item in a scenario is determined by the inventory in the previous period plus the quantity ordered from the warehouse. From this inventory we subtract the demand for the item if that item is needed in combination with its successor.

$$I_{its} = I_{i(t-1)s} + Q_{it} - \sum_{j \in A} a_{ij} X_{j(t-p_{js})} - d_{it} \qquad \forall i, t, s \qquad (2)$$

Constraint (3) is the capacity constraint related to the unpacking areas. If a setup for an item occurs, the quantity of the item times the setup time of the item must be smaller or equal to the capacity for its designated unpacking area. Here the auxiliary variable related to the overtime indicated by O_{mt} comes in to place to allow the model to find a feasible solution.

$$\sum_{i \in N} Q_{it} \cdot st_{im} \leq c_{mt} + O_{mt} \qquad \forall m, t \qquad (3)$$

Constraint (4) forces a setup if a the decision is made to order a certain quantity of an item in a certain period. If a quantity is ordered, the binary setup variable must become 1 to satisfy the constraint as it then multiplies with a large number. Note that this only holds for every item in the set of purchase items.

$$Q_{it} \leq B \cdot y_{it} \qquad \forall i, t \qquad (4)$$

Constraint (5) forces X_{jt} to be 1 if a parent item is ordered in a certain period in the same way as Constraint (4) does. By forcing this binary variable instead of just using Q_{jt} in our inventory balance constraint we remove the multiplication of materials that have a parent-component relation.

$$Q_{jt} \leq B \cdot X_{jt} \qquad \forall j, t \qquad (5)$$

The delivery of materials to the production area can start when an external customer has placed an order and issues a desired delivery date. The items used for the project are specific to the project, therefore the starting inventory, or the inventory in period 1 should be 0 and the ending inventory should be 0 as well. This is done in constraint (6).

$$I_{i1s} = I_{iTs} = 0 \qquad \forall i \qquad (6)$$

The variables related to the order/production quantity can only take on an integer value that is larger than or equal to 0.

$$Q_{it}, I_{its}, O_{mt} = INT \qquad \forall i, t, \text{s,m} \qquad (7)$$

The setup variable is a binary variable that is 0 if there is no setup for an item in a period and switches to 1 otherwise.

$$y_{it}, X_{it} \in \{0,1\} \qquad \forall i, t \qquad (8)$$

Having presented the sets, parameters and variables and elaborating on the objective function followed by the constraints set for this project we now have the complete model of the Multi-Level Capacitated Lot-Sizing Problem for the production area of Systems-2.

4.5 Solution approach

In this paragraph we introduce the solution approach that is used in this research. The solution of our MIP model is a setup plan. This setup plan describes for every item in which period a setup must take place. In our solution approach, we destroy part of the setup plan and insert this partially destroyed solution into the MIP. The MIP optimizes the destroyed part of the setup plan. This way, instead of only using the MIP to find a solution, we let the MIP iteratively solve small parts of the solution to optimality.

4.5.1 Destroy and repair algorithm

In the literature review we described the FO approach developed by Chen H. (2015). The solution approach that is used for this research is inspired by this Fix-and-Optimize algorithm, but has some modifications. This solution approach make use of a hybrid system. This means that we make use of an heuristic that also uses the MIP model. Before we can start the algorithm, we first have to find a good initial solution. In the original FO approach, this is done by fixing the values of all setup variables y_{it} to 1, so for every item in every period a setup takes place. However, this results in a solution that cannot be considered as 'good' because it leads to excessive setups and high setup costs, making it an inefficient starting point. Instead, we obtain an initial solution by applying a LP-relaxation of the MIP model. This means that we remove the integrality constraints for all of the decision variables. This relaxation technique transforms an NP-hard problem such as ours in to a related problem that can be solved more efficiently. However, the solution that is found in this LP-relaxation is not feasible and cannot directly be used for our algorithm. To create a feasible starting solution, we set $y_{it} = 1$ if the relaxed solution assigns it a value greater than 0; otherwise, we set $y_{it} = 0$. This ensures that only necessary setups are included in the initial feasible solution. Unlike the original FO method, which initialized all setups, this approach uses insights from the LP-relaxation to identify promising setups for the initial solution.

Now that we have constructed an initial solution, we can apply the fix and optimize approach to try to improve the initial solution. The solution in this case is the setup plan, so the values for every setup variable y_{it}. The idea of the FO approach is that a part of the solution is destroyed by removing the fixed values for a selection of setup variables y_{it}. By destroying a part of the solution, we create a subproblem SP that focusses specifically on optimizing the destroyed part of the solution. For the optimization of this subproblem is we make use of the MIP.

To destroy a part of the solution we first select a pair (i, t) for setup variable y_{it}. This pair (i, t) serves as a starting point for the selection of setup variables to be destroyed. This can be done randomly, but also based on some logic, such as the pair with the highest setup or highest inventory cost. In Chapter 5, a numerical experiment will show how the selection of this pair (i, t) for setup variable y_{it} will affect the algorithm. The selection of the other setup variables y_{it} to be destroyed in the solution must be done carefully, because the value of a setup variable may lead to the change of the value of another setup variable if they are coupled with each other through a constraint. For instance, if we want the setup for an item to take place in another period, there must be capacity available in that period and the parent item or predecessor must take place in another period. Because due to the inventory balance constraint, a predecessor must be delivered at an earlier point in time than its parent. We must also allow the model to place a setup in another period, this can only be done if the value of the setup variable in other periods is also destroyed. We call these type of setup variables 'interrelated'. The setup variables to be destroyed are selected based on their interrelatedness with other setup variables to ensure that the model has room to find new values for the setup variables. We consider a setup variable to be interrelated if one of the following three conditions holds:

1. The setup variable is interrelated with a setup variable for the same item but in the previous or next period.
   $i' = i \text{ and } t' \in \{t-1,\ t,\ t+1\}$
   Example: if $(i, t) = (2,5)$ then $y_{2,5}$ is interrelated with $y_{2,4}$, $y_{2,5}$ and $y_{2,6}$
2. The setup variable is interrelated with a setup variable for the predecessor or successor of the item in the same period.
   $i' \in P_i \cup S_i \text{ and } t' = t$
   Example: For this example we consider that item 1 is the predecessor of item 2 and item 3 is the successor of item 2. if $(i, t) = (2,5)$ then $y_{2,5}$ is interrelated with $y_{1,5}$ and $y_{3,5}$.
3. The setup variable is interrelated with a setup variable for an item that must be unpacked or cleaned in the same area and in the same period.
   $i' \in N_{m_i} \text{ and } t' = t$
   Example: For this example we consider that item 2 and item 7 both use capacity for the unpacking area. if $(i, t) = (2,5)$ then $y_{2,5}$ is interrelated with $y_{7,5}$.

Now that the conditions for a setup variable to be interrelated with another setup variable are presented, we move on to the level of interrelatedness. In the small examples mentioned in the definition of condition the focus is on the variables that are interrelated with $(i, t) = (2,5)$. The setup variables directly linked to the first selected setup variable (i, t) based on the above conditions are called 1-interrelated. The selected 1-interrelated setup variables form the set IR^1. In Figure 21 a visualization of the selection of the 1-interrelated variables is shown.

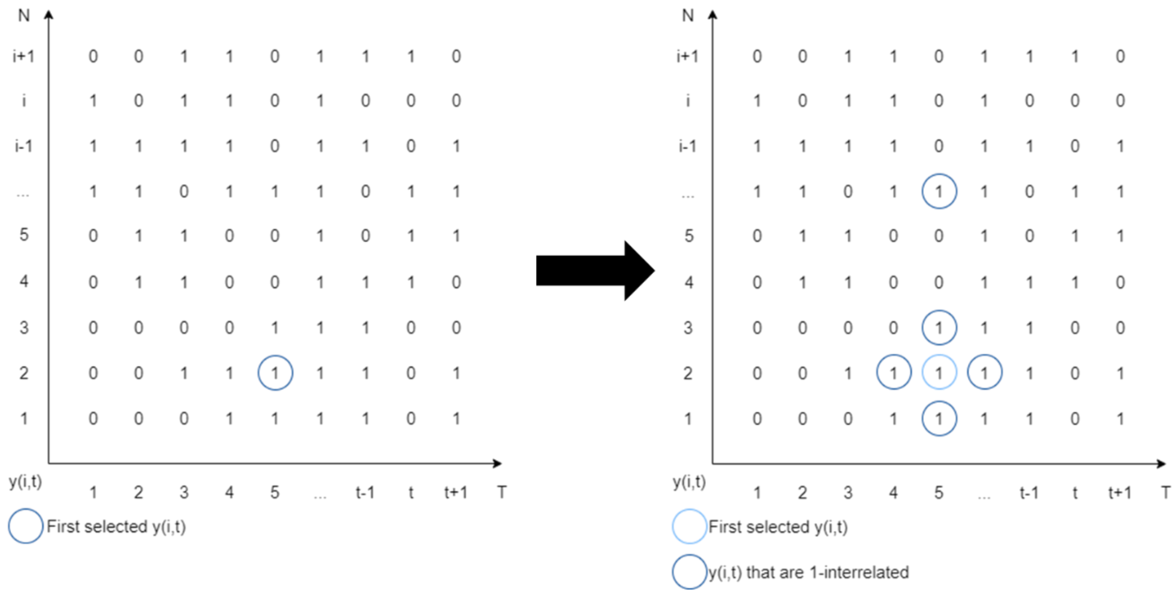


Figure 21: Selection of setup variables that are 1-interrelated

To increase the degree of destruction we look at the setup variables that are interrelated with any setup variable in IR^1. These setup variables form the set IR^2. In Figure 22, a visualization of the selection of 2-interrelated setup variables is given. This Figure, as well as Figure 21 serve the purpose of illustrating which variables are 1- and 2-interrelated, therefore a distinction is made between the first selected setup variable y_{it}, the setup variables that are 1-interrelated and the setup variables that are 2-interrelated. In practice the set of 2-interrelated setup variables also contains the first selected setup variable and the setup variables that are in IR^1 because every setup variable is interrelated with itself through condition 1.

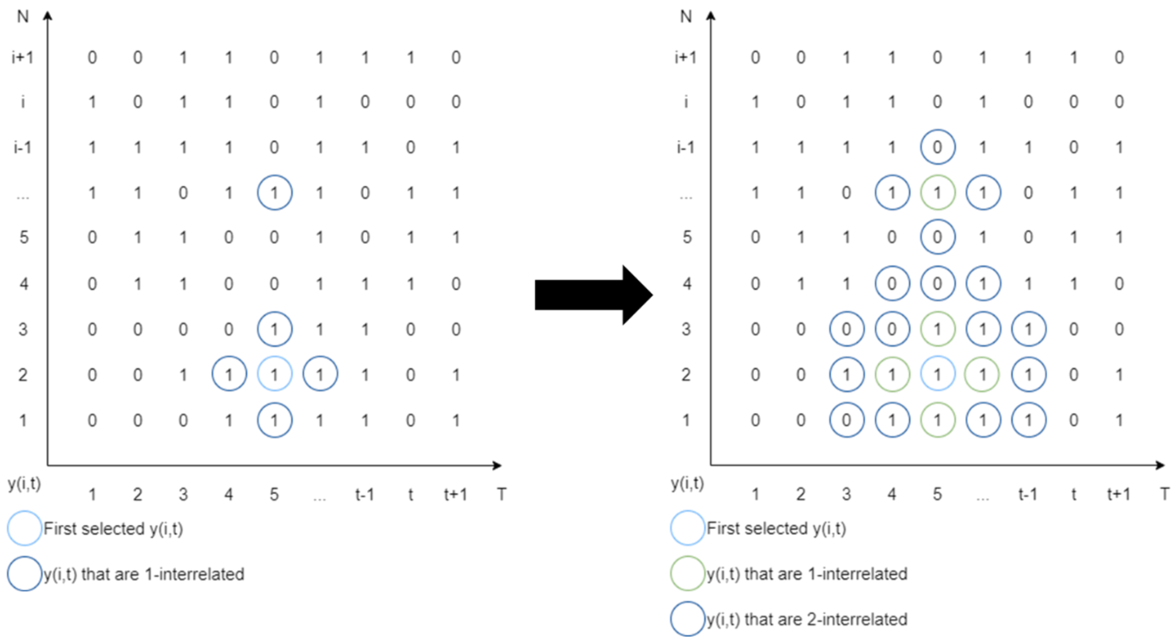


Figure 22: Selection of setup variables that are 2-interrelated

From IR^2 we can still increase the degree of destruction by selecting the setup variables that are interrelated with any setup variable in IR^2. We call these variables l-interrelated. For any integer $l > 1$, two setup variables are l-interrelated if there exists a chain of 1-interrelated variables, forming the set IR^l. As the value of l grows, the number of selected setup variables grows exponentially. This leads to a larger degree of destruction but also impacts the computation time. Therefore, l functions as a control parameter in the algorithm to balance computational efficiency and solution quality. Chen H. (2015) is aware of this trade-off and has tested the FO algorithm with $l = 1, 2, 3$ and found that $l = 2$ results in the best tradeoff. Since this research is differs from the research of Chen H. (2015), we will experiment with this control parameter in Chapter 5. This experiment starts with $l = 2$ as control parameter for the algorithm for every condition. Then the effect of setting $l = 1$ and setting $l = 3$ for a single condition will be analyzed for every condition separately. Based on these experiments a final decision will be made on the control parameter l.

Having established the initial solution and identified the setup variables to be destroyed, we now proceed with the optimization. Removing these setup variables leaves us with a partial solution from the initial solution which we then insert in the MIP. This MIP includes an additional constraint:

$$y_{it} = PartialSolution_{it} \qquad \forall i, t \qquad (9)$$

Here $PartialSolution$ contains what remains of the solution after destroying a part of the solution. this additional constraint ensures that every value of y_{it} is fixed to the value of y_{it} in the partial solution. The MIP uses this partial solution as a foundation to reconstruct a new, feasible solution. This new, feasible solution is then compared to the initial solution and if the objective value of the new solution is better, we accept this solution, otherwise we reject the solution. If we accept the new solution, the current best objective and current best setup plan is updated. All these steps form 1 iteration. In the next iteration we again destroy the current best solution and reoptimize this solution by inserting the partial solution in the MIP.

After iteratively destroying and repairing the solution for some time, it can occur that the structure of the solution is formed in such a way that we cannot find improvements anymore. This means we are stuck in an optimum. Unless we have explored every optimum in the solution space, we can say whether it is a local or a global optimum. However, if we are stuck in a local optimum, we want to escape this optimum by diversifying to a different search area. We do this by destroying the solution with the same destroy heuristic as mentioned above. However, instead of checking if the objective value of this solution is better than the current best objective value, we now accept the solution. Even if it's worse than our current best solution. By accepting the new solution we aim to change the solution structure such that we escape the local optimum.

To prevent that we explore a region that we have previously visited, we include a tabu list that stores the first selected pair (i, t) after a diversification. We include a tabu list because in Chapter 5 we experiment with different diversification strategies. Some of these strategies select the first pair based on a greedy approach. This can result in selecting a pair that is previously selected which is something we do not want to happen. In Appendix B the pseudo code of the destroy and repair algorithm is given. Figure 23 presents a flow chart of the full destroy and repair algorithm.

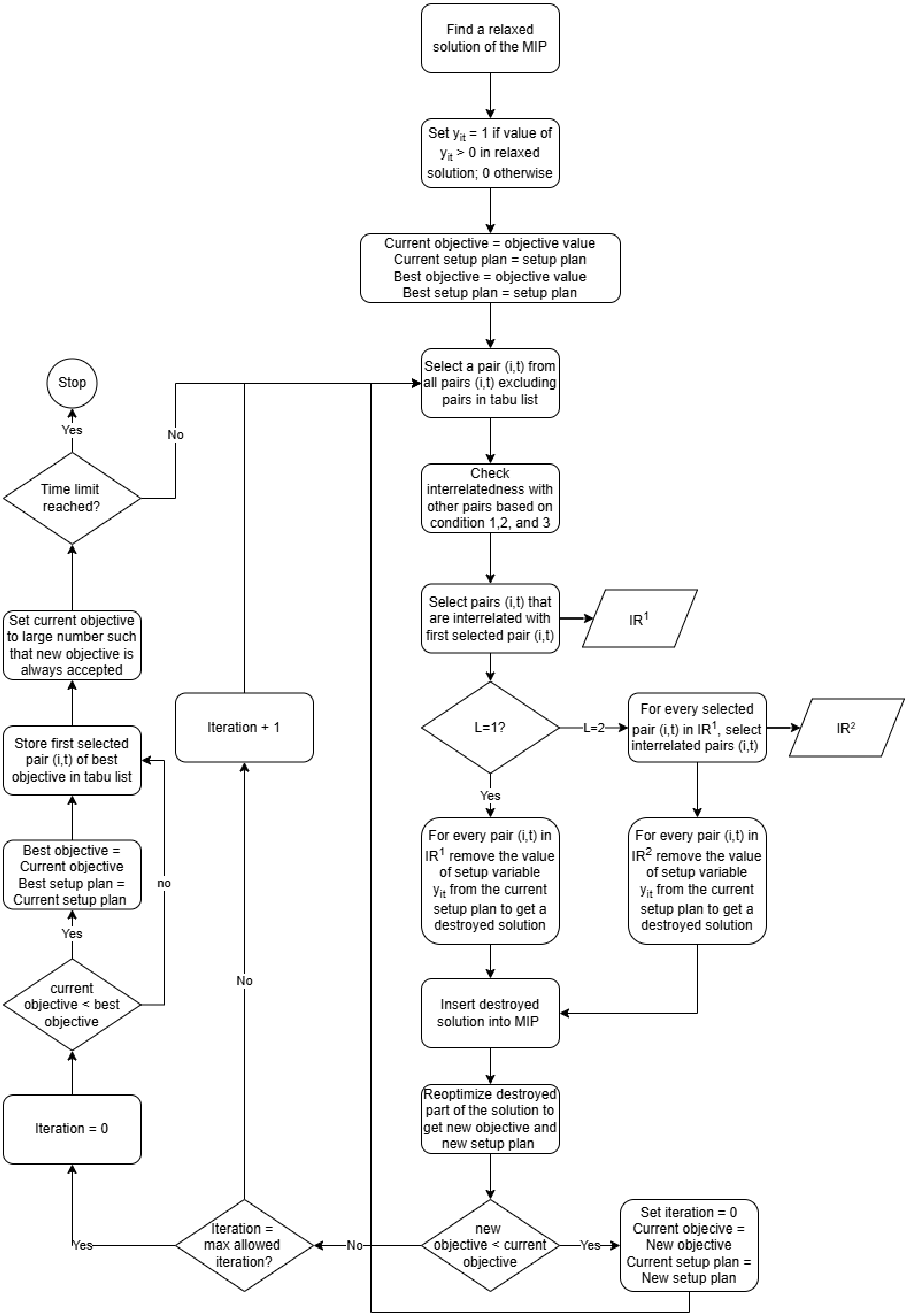


Figure 23: Flow chart of the destroy and repair algorithm

4.5.2 Running the model

First it is needed to implement the mathematical model into a program that is able to compute outcomes for the MLCLSP. As mentioned in the introduction, a potential risk for this research is the selection of software that is used to develop and running the model as it can cause high investment costs for Systems-2. Therefore, this model will be implemented in Python – which is a free to use programming language – by making use of the Python-MIP package. The Python-MIP package is a collection of Python tools for the modeling and solution of Mixed-Integer Linear programs. The default installation includes the COIN-OR Branch and Cut solver – CBC, which is a highly configurable solver and is, according to Görner et al. (2021) among the fastest open source MIP solvers. It also works with the state-of-the-art Gurobi MIP solver. But as mentioned, for Systems-2 it is important that it is free to use. Therefore we use the default installation, COIN OR CBC. With this setup, the model can be used after this research. Another reason why this is important is the fact that this research limits to only two modules. By using a free-to-use software and solver it enables Systems-2 to run the model for the remaining modules. Since we are dealing with a large complex problem, we have also included additional statements to make optimal use of the PC and to let the MIP model focus more on finding feasible solutions. We have added the following two statements:
Model.threads = -1
Model.emphasis = SearchEmphasis.FEASIBILITY.value
The first statement relates to the number of threads to be used when solving the problem. A value of 0 uses the solver default configuration, -1 uses the number of available processing cores. The second statement relates to the search emphasis. Using this statement, we use a more aggressive search for feasible solutions instead of optimal solutions. This is because we are more interested in finding feasible solutions in each iteration of our algorithm.

4.6 Summary

We have made the necessary assumptions on the modules to evaluate, the setup times, the available capacity and the uncertain timing of demand to simplify the problem while still making the model represent the actual situation as much as possible. The input data for the model was then prepared by constructing a Gozinto for all parent-component relations. Also other input parameters such as holding cost, setup cost and setup time had to be prepared for every item. After combining components of the two MLCLSP models presented in the literature review, a few additions and modifications still had to be made to accurately represent the problem situation at Systems-2. After formulating the model, we have implemented the model in Python, where we make use of the Python-MIP package. The default installation of this package includes the COIN-OR Branch and Cut solver – CBC, which is a highly configurable solver and free to use. In addition to the MIP implementation in Python, a destroy and repair approach is implemented to improve the solution given by the MIP. This improved solution will be analyzed in Chapter 5.

5 Analysis of results

This chapter presents the results of the MLCLSP model that is specifically designed for the problem situation at Systems-2. In this chapter we answer the following research question:

How can a policy reduce the WIP inventory at the production area?

In Section 5.1 we test different levels of interrelatedness between setup variables to find an optimal configuration for our algorithm. In Section 5.2 we experiment different combinations of local search and diversification strategies. In Section 5.3 we compare the algorithm to a simplified version to demonstrate the added value of the decision made during the development of the algorithm. In Section 5.4 we compare the new found setup policy to the current setup policy for both modules. Section 5.5 we perform a sensitivity analysis to show how consistent and robust our model is.

5.1 Level of interrelatedness

In Section 4.5.1, we present the method for destroying part of a solution such that we can re-optimize the remaining partial solution. Since the change of the value of a setup variable may lead to the change of another setup variable if they are coupled with each other through constraints, setup variables are selected based on their interrelatedness with other setup variables. However, the number of setup variables removed from the solution has a significant impact on the performance of the MIP. If the number of selected setup variables is too small, we have less diversification, and the effect of a large neighborhood is lost. On the other hand, if the number of setup variables removed from the solution is too large, we have less intensification and risk finding many poor-quality solutions. In our FO approach, the number of selected setup variables to remove from the solution is dependent on the level of interrelatedness (l) between setup variables. Chen H. (2015) studied the performance of the FO approach with different levels of interrelatedness between setup variables and concluded that $l = 2$ provides the best results when applied uniformly across all conditions. This means that we first check the variables that are interrelated with the first selected pair to get IR^1 and then for every pair in IR^1 we again check the variables that are related to get IR^2. $L = 1$ results in IR^1 and $l = 2$ results in IR^2. So $l = 2$ for all conditions below:

- Condition 1: The setup variable is interrelated with a setup variable for the same item but in the previous or next period.
- Condition 2: The setup variable is interrelated with a setup variable for the predecessor or successor of the item in the same period.
- Condition 3: The setup variable is interrelated with a setup variable for an item that must be unpacked or cleaned in the same area and in the same period.

However, for this research we aim to investigate whether different levels of interrelatedness per condition can improve performance. To analyze this, we design an experiment that systematically varies l across the interrelatedness conditions. By adjusting l for one condition at a time while keeping the others fixed, we can isolate the effect on the number of selected setup variables and the solution quality. Additionally, cases are tested where l is increased by 1 for two conditions simultaneously. This approach provides insights into whether a uniform $l = 2$ remains optimal or if a more tailored configuration enhances performance.

In the experiment we refer to a configuration by: (Condition 1/ Condition 2/ Condition 3).

- The experiment starts with a uniform $l = 1$ for every condition, or (1/1/1);
- l is increased by one for every condition separately: (2/1/1), (1/2/1), (1/1/2);
- l is increased by one for two separate conditions: (2/2/1), (2/1/2), (1/2/2);
- Repeat this procedure starting from a uniform $l = 2$ for every condition;

- At last also test a configuration with a uniform $l = 3$.

The goal of this experiment is to investigate whether using different interrelated levels per condition improves the performance over using a uniform level. To evaluate the performance, we are interested in three indicators: the number of setup variables that are selected for the destruction of the solution (size of the subproblem), the solving time in seconds, and the improvement of the solution with respect to the initial solution. The initial solution is constructed by solving the LP relaxation for 60 seconds and fixing the values of the setup variables to 1 if the value of the setup variable in the LP relaxation is greater than 0, and 0 otherwise. These fixed values of the setup variables are then inserted in the MIP to find a feasible initial solution. In this experiment we are interested in the effect of interrelatedness on the first selected pair (i, t). Therefore, we select a first pair (i, t) that has an interrelation with another pair (i, t) through every condition. From this initial pair (i, t) we select setup variables based on their interrelatedness compliant with the configuration used. The initial solution is then destroyed by removing the values of these selected setup variables. This destroyed solution is then inserted into the MIP which repairs the solution by optimizing this selection of setup variables. For every configuration the maximum solving time has been set to 60 seconds and the same initial solution and same first selected pair (i, t) are used within a single run, to ensure reliable comparisons across all configurations. The full experiment is repeated 15 times, each with a different seed value for the MIP model (incremented by 1 each time). This way every experiment starts with a new initial solution and a new pair (i, t) to ensure that the results are not influenced by a specific instance and to improve the overall robustness of the conclusions. For this experiment we use the input data of Module 4. The set of items N and set of periods T of Module 4 is significantly smaller than those of Module 1. This allows us to use a time limit of 60 seconds per configuration. Due to the problem size of Module 1, it can take up to 5 minutes to find a subproblem with a configuration that selects a large number of setup variables.

Table 4: Experiment with configurations consisting of different levels of interrelatedness per condition

		Number of variables			Solving time (seconds)			Improvement of initial solution in %		
Experiment	Configuration	LB	AVG	UB	LB	AVG	UB	Best	AVG	Worst
Uniform 1	(1/1/1)	5.00	40.33	56.00	2.33	3.21	5.98	0.00%	0.00%	0.00%
Increase 1 condition to 2	(2/1/1)	11.00	117.00	164.00	2.30	4.77	6.60	-2.86%	-2.06%	0.00%
	(1/2/1)	579.00	597.00	597.00	3.93	56.25	62.18	-5.43%	-2.82%	0.00%
	(1/1/2)	2.00	43.53	64.00	2.15	4.67	6.38	0.00%	0.00%	0.00%
Increase 2 conditions to 2	(2/2/1)	581.00	581.00	581.00	3.80	54.92	71.09	-5.43%	-3.03%	0.00%
	(2/1/2)	11.00	123.07	175.00	2.45	4.76	6.09	-2.86%	-2.06%	0.00%
	(1/2/2)	579.00	597.00	597.00	3.82	56.69	61.54	-5.43%	-2.95%	0.00%
Uniform 2	(2/2/2)	581.00	581.00	581.00	3.93	55.22	70.97	-5.43%	-2.99%	0.00%
Increase 1 condition to 3	(3/2/2)	967.00	967.00	967.00	61.06	61.32	62.31	-6.50%	-3.03%	0.00%
	(2/3/2)	965.00	965.00	965.00	62.11	62.92	72.08	-5.00%	-3.06%	0.00%
	(2/2/3)	13.00	60.40	85.00	1.97	3.56	5.51	0.00%	0.00%	0.00%
Increase 2 conditions to 3	(3/3/2)	966.00	966.00	966.00	62.07	63.61	72.47	-4.75%	-2.57%	0.00%
	(3/2/3)	581.00	587.80	606.00	55.70	60.88	61.98	-3.71%	-0.40%	0.00%
	(2/3/3)	965.00	965.00	965.00	62.04	62.34	63.31	-6.61%	-2.99%	0.00%
Uniform 3	(3/3/3)	967.00	967.00	967.00	62.13	63.05	72.25	-6.62%	-3.05%	0.00%

Table 4 presents the number of selected setup variables, the solving time, and the improvement of the solution in percentages for every configuration. For the number of variables, the lower bound, upper bound, and the average value are shown. For the improvement in percentages, the best, average and worst improvement are presented. Where the best improvement is the highest reduction of the objective value compared to the initial solution. Configuration (2/1/1) and (2/1/2) are the only two configurations that consistently achieve low solving times while also improving the initial solution. Configurations (1/1/2) and (2/2/3) also show consistently low solving times but are not able to improve the initial solution. This suggests that selecting setup variables primarily based on the third condition does not contribute to improving the solution quality. This is probably because the variables that are interrelated through condition 3 are not interrelated with itself. So if we check the interrelatedness of setup variables only through condition 3 for every pair in IR^1 or IR^2 we do not select the variables that are in IR^1 or IR^2 which we do when we are checking the interrelatedness through condition 1. This will leave us with a lower number of setup variables that are interrelated with variables that are not selected. Therefore the effect of selecting variables based on their interrelatedness is lost.

It is interesting to see that the configurations that select many variables such as (3/2/2), (2/3/2) and (3/3/3) are able to find the best improvements compared to the initial solution. However these configurations almost always use the available computation time completely. Due to the high solving times of these configurations, we do not select one of these configurations for our model. These configurations however could be interesting for other applications.

In configurations where $l = 2$ or $l = 3$ is applied for the second condition, the subproblem tends to become too large, making it too complex for the MIP to solve. In most cases the time limit is reached, and in some cases the model is not able to improve the initial solution. Since increasing l for condition 1 while keeping condition 2 at $l = 1$ yields promising results, an additional experiment is conducted. For this additional experiment, we fixed $l = 3$ for condition 1, and increase l for condition 3 by 1, starting from $l = 1$.

Table 5: Additional experiment for promising configurations

		Number of variables			Solving time (seconds)			Improvement of initial solution in %		
Experiment	Configuration	LB	AVG	UB	LB	AVG	UB	Best	AVG	Worst
Promising config.	(2/1/1)	11.00	117.00	164.00	2.30	4.77	6.60	-2.86%	-2.06%	0.00%
	(2/1/2)	11.00	123.07	175.00	2.45	4.76	6.09	-2.86%	-2.06%	0.00%
Conditon 1 = 3. Condition 3 + 1	(3/1/1)	12.00	142.27	344.00	0.94	3.25	6.21	-2.89%	-1.03%	0.00%
	(3/1/2)	12.00	145.13	357.00	1.04	3.12	5.10	-2.89%	-1.03%	0.00%
	(3/1/3)	12.00	153.20	358.00	0.92	3.03	5.41	-2.89%	-1.03%	0.00%

Table 5 shows the results of the additional experiment. As expected, the number of selected setup variables increases. Although the best improvements achieved by the additional configurations are better than configurations (2/1/1) and (2/1/2), the average improvement is lower. This indicates that the MIP is more often not able to improve the initial solution, which is often accompanied by shorter solving times. Based on the original and additional experiment, configuration (2/1/2) is selected as it offers the best tradeoff between solution improvement and solving time for this project.

5.2 Algorithm experiment

Now that the optimal configuration for the level of interrelatedness has been determined, the focus shifts to experimenting with the algorithm itself. The removal of setup variables from the solution starts by selecting a pair (i, t). Chen H. (2015) proposes that this pair should be selected randomly from $N \times T$ with the same probability for each element in $N \times T$. Although the remaining setup variables are selected based on their interrelatedness with the initial pair, this random approach may lead to inefficient removals. Therefore, we investigate the effect of selecting the first pair for a destroy step using a more greedy approach. The objective function of the MLCLSP contains three cost components: setup cost, inventory cost and overtime cost. However, overtime cost are not always present in the solution because they only arise when capacity constraints are violated. Since setup cost and inventory cost are always present in a solution, they provide two viable alternatives for a greedy selection strategy for the initial pair (i, t).

1. First selected pair (i, t) is the pair for which the setup variable in the current best solution has the highest setup cost. This approach aims to re-optimize the most expensive setup decision in the current solution by using it as the starting point for the setup variable removal procedure.
2. First selected pair (i, t) is the pair for which item i has the highest inventory cost in period t. This inventory cost is used to identify where in the solution the holding cost are relatively high, with the aim of re-optimizing that part of the solution. Once this pair is selected, the associated setup variable is used as the starting point for the setup variable removal procedure.

In Section 4.5.1, we discussed the need to diversify the search to avoid getting stuck in a local optimum. Chen H. (2015) suggests selecting a new random pair (i, t) to diversify. While this approach helps explore the solution space, it may also lead to many poor-quality solutions. To mitigate this, we investigate an alternative diversification strategy that introduces a more targeted selection process. More specifically, we explore whether selecting the pair (i, t) for diversification using a greedy approach improves performance. Since selecting a pair based on the highest setup cost or the highest inventory cost provides a structured way to guide local search, we also apply this to diversification to examine their impact on the performance. In this experiment, we refer to a combination by (local search strategy / diversification strategy). The following five separate combinations are tested:

- (Random / Random)
- (Setup cost / Random)
- (Inventory cost / Random)
- (Setup cost / Inventory cost)
- (Inventory cost / Setup cost)

Instead of testing all 10 possible combinations, these 5 combinations are selected to contrast fully random strategies with more guided, cost-based strategies. It enables us to investigate the effect of using a greedy local search compared to a random local search strategy. In addition to that, it also enables us to investigate the effect of diversifying the search area by moving to a neighborhood that is potentially more promising, as it is selected base on cost information rather than random.

To evaluate the performance, we are interested in three indicators: the best objective value found per combination, the average objective value per combination, and the progression of the best found solution over iterations. Here, we also construct the initial solution by solving the LP relaxation for 60 seconds and fixing the values of the setup variables to 1 if the value of the setup variable in the LP relaxation is greater than 0, and 0 otherwise. These fixed values of the setup variables are then inserted in the MIP to find a feasible initial solution. In every run, we use the same initial solution for each of the combinations to make reliable comparisons. For all the above mentioned combinations we use the following settings:

- If there is no improvement in the solution after 5 iterations, we diversify the search space.
- The length of tabu list is set to 10.
- Time limit for improving a solution is set to 60 seconds.
- Time limit for one run is set to 10 minutes.
- We run each combination 10 times.

Table 6 presents the best objective value found and the average objective value and variation over 10 runs for each combination. Based on the results of the experiment, the combination (Random / Random) is selected for the remaining part of this research. Although both (Random /Random) and (Setup cost / Random) were able to reach the best-known solution in 3 out of 10 runs, the (Random / Random) combination showed a lower average objective value and a smaller variation across all runs. This indicates that this combination has a more stable and consistent performance, making it the preferred option. What is interesting to see, is that the combinations with a random diversification strategy all perform better than the combinations that diversify based on logic. This has probably to do with the fact that a random diversification strategy is able to explore more neighborhoods and is able to escape the local optimum. Diversifying based on the inventory cost or setup cost of the local optimum may not disrupt the solution structure enough. This keeps the search too close to the local optimum, making it harder to escape to a different, potentially better region of the solution space.

Table 6: Experiment results for different combinations of local search strategy and diversification strategy

Run	IC/RND	IC/SC	RND/RND	SC/IC	SC/RND
1	7441.51	7416.25	7358.28	7443.37	7358.28
2	7416.25	7433.58	7416.25	7433.58	7416.25
3	7416.25	7423.45	7416.25	7443.37	7416.25
4	7436.17	7541.29	7416.25	7541.29	7416.25
5	7388.17	7443.37	7358.28	7443.37	7358.28
6	7452.27	7449.08	7486.09	7361.55	7514.17
7	7454.30	7526.60	7439.61	7624.52	7519.40
8	7388.17	7443.37	7416.25	7443.37	7416.25
9	7388.17	7443.37	7358.28	7443.37	7358.28
10	7499.48	7435.84	7416.25	7436.17	7416.25
Average	7428.07	7455.62	7408.18	7461.40	7418.97
Variation	111.32	125.04	127.81	262.98	161.13
Best	7388.17	7416.25	7358.28	7361.55	7358.28

In addition to the experiment results for the different local search and diversification strategies it is interesting to see how the objective value progresses over time. Therefore, an additional run has been performed to register the progression of the solution quality over the iterations.

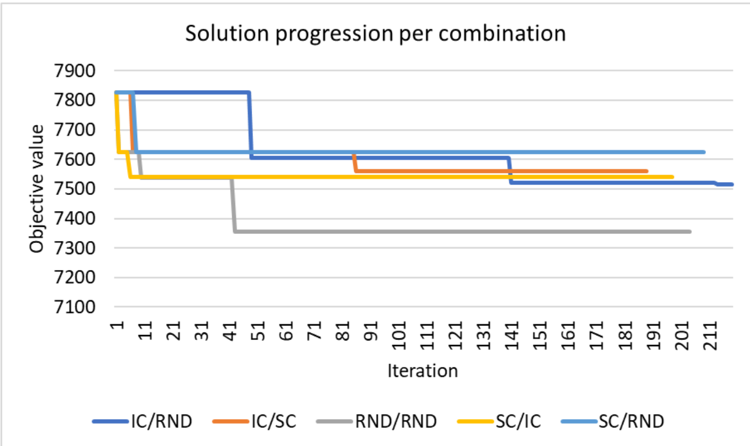


Figure 24: Solution progression for different combinations of local search strategy and diversification strategy

In Figure 24 the progression per combination is shown. In this Figure it can be seen that the RND/RND combination also outperforms the other combinations in terms of solution progression. It is able to find the best solution – compared to the other combinations – after approximately 45 iterations. The combinations SC/RND, and SC/IC also converge fast but at worse solutions. The combinations IC/RND and IC/SC seem to be able still find improvements after a high number of iterations, especially the combination IC/RND. A possible explanation for this is that the Random diversification strategy is able to explore more regions of the solution space.

5.3 Algorithm performance

Now that we have finalized the configuration the algorithm, we aim to evaluate its performance. To demonstrate the added value of the decisions made during the development of the algorithm, we compare our algorithm to a simplified version of the algorithm. Table 7 provides an overview in the differences between our algorithm and the simplified version of the algorithm.

Table 7: Overview of differences between our algorithm and uncomplex version of the algorithm

Aspect	Our algorithm	Simplified algorithm
Construction of initial solution	Solve the MIP relaxation and set the value for every setup variable y_{it} to 1 if the relaxed value of $y_{it} > 0$; 0 otherwise.	Set the value of every setup variable y_{it} to 1 and insert this into the MIP to get an initial feasible solution.
Selection of the setup variables y_{it} from the setup plan to destroy and repair	Select setup variables y_{it} that are interrelated based on three conditions.	Select 100 random setup variables y_{it}.
Variable neighborhood search	Yes	No

We run this simplified version of the algorithm for 30 minutes and apply it to Module 4. The best solution that the simplified algorithm is able to find is an objective value of 7719. This is worse than all the objective values that we found during the experiment with different local search and diversification strategies. Figure 25 shows the solution progression of the simplified algorithm. It shows that the construction of the initial solution by setting the value of all the setup variables to 1 results in an initial solution with a high objective value (+- 31000). This, in combination with the random selection of setup variables to be destroyed, makes it hard and time consuming to really reduce the objective value. Since the improvements per iteration are small, the need for a variable neighborhood search becomes redundant.

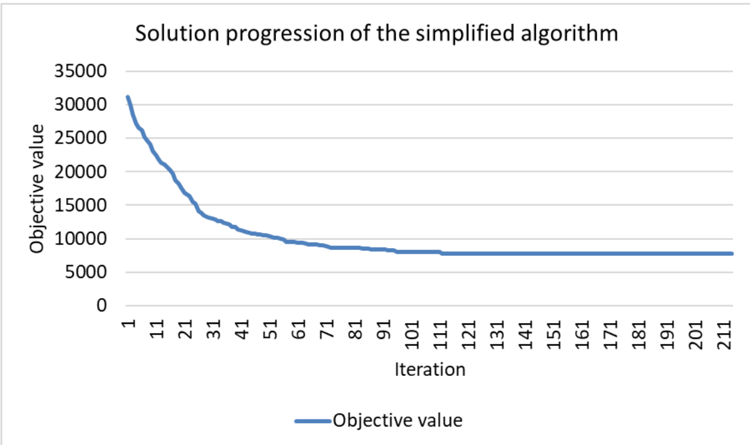


Figure 25: solution progression of the simplified algorithm

Having compared our algorithm to a simplified version of the algorithm shows that our algorithm is able to construct a good initial solution. It also shows that our algorithm is able to find a better solution by selecting the setup variables to be destroyed based on the interrelatedness between variables and by making use of a variable neighborhood search.

5.4 Comparison with current setup policy

In this paragraph, the model is applied to generate solutions for both Module 1 and Module 4. These solutions consist of newly developed setup policies. To evaluate the impact of these new policies, a comparison is made with the current setup policies used by Systems-2. This is done by implementing the current setup policies for both modules into the MIP model and comparing the setup cost, inventory cost and capacity usage with those of the newly developed setup plans. The current setup policy is based on the current delivery moments that are used for both separate modules. This information is available in BAAN and is gained from the Production Assistant of Systems-2.

5.4.1 Module 1

In the current situation, Systems-2 uses a setup policy for Module 1 in which there are four separate delivery moments: period 1, 3, 9, 13 and 17. This setup policy is inserted in the MIP by fixing the setup variables for every item in the other periods to 0. With this setup policy however, the MIP is not able to find a solution at all. Therefore the setup policy is modified by extending the setup period by 1 for every delivery moment. This results in the following current setup policy for the MIP: period 1, 2, 3, 4, 9, 10, 13, 14, 17, 18. This way the structure of the setup policy remains and the MIP is able to find a solution.

5.4.1.1 Setup cost

Figure 26 shows the total setup cost per period of the current and new setup policy respectively. Only setups that consume capacity and therefore incur setup cost are relevant, as these are setups for materials that need to be delivered to the production area. The new setup policy allows delivery moments in more periods compared to the current policy. This is caused by the fact that in the new setup policy, the capacity usage per period is taken into consideration. These additional delivery moments do however result in a higher total setup cost of €8487.14 compared to a total setup cost of €7684.19 in the current policy. The difference in costs is due to the assumption that whenever a setup for an item occurs, the associated cost corresponds to the highest quantity required for that item. In the new policy, some items are delivered across multiple periods resulting, in smaller quantities to be shipped each time. However the setup cost remains based on the full quantity, as if the total quantity is shipped.

We can see that in period 2,3, and 4 the setup costs are at least twice as large as in the new policy, other than that, a large difference can be seen in period 18. Where there are no setup cost in the new policy compared to a setup cost of €1200 in the current policy. The setup cost in periods 9 and 10 and periods 13 and 14 show similar behaviour but here we must take into considerations that every delivery moment is extended by one period. If we aggregate the cost of the added periods 2, 4, 10, 14 and 18 and spread the cost over the original delivery periods in the current setup policy, the setup cost will become significantly larger in these periods.

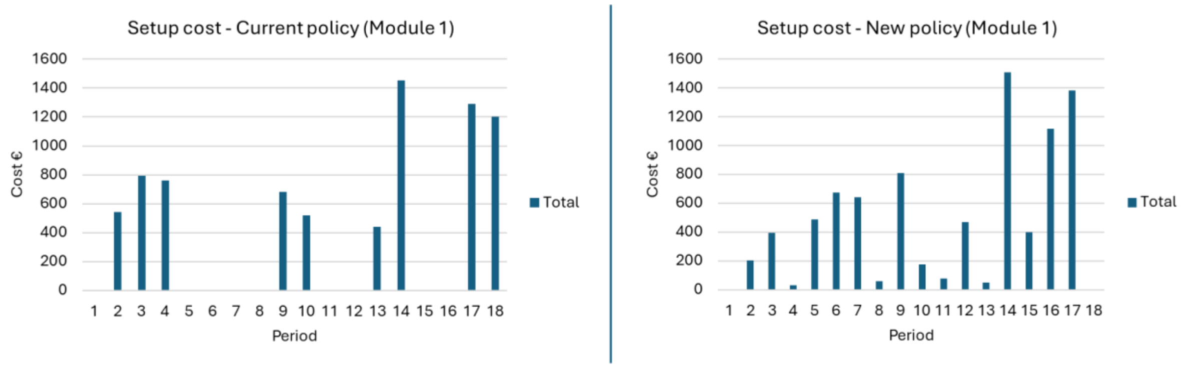


Figure 26: Setup plan of the current policy and new policy of Module 1

5.4.1.2 Inventory cost

Figure 27 shows the inventory cost per scenario of the current and new setup policy respectively. Here the effect of the new setup policy becomes visible. In the current setup policy we have a consistently high inventory cost except for period 10, 11, and 12 where there are still inventory cost but much lower, resulting in a total inventory cost of €203,675.63. Even in the scenario where items are late, there are still inventory cost. In practice, when items arrive late, the production is desperately waiting for the items arrive. The items will then directly be put use to resume the production, resulting in the items not being inventory at all. This illustrates that with the current setup policy many items are delivered into the production area while they are not yet needed as when they arrive late, they are still being stored. In the new policy, there are also inventory cost, but significantly less compared to the current setup policy with a total inventory cost of €88,285.80 for the new setup policy. The inventory cost per scenario are also logical, as there are higher inventory cost in the scenario that items are delivered early, and there are almost no inventory cost in the scenario where items are delivered late.

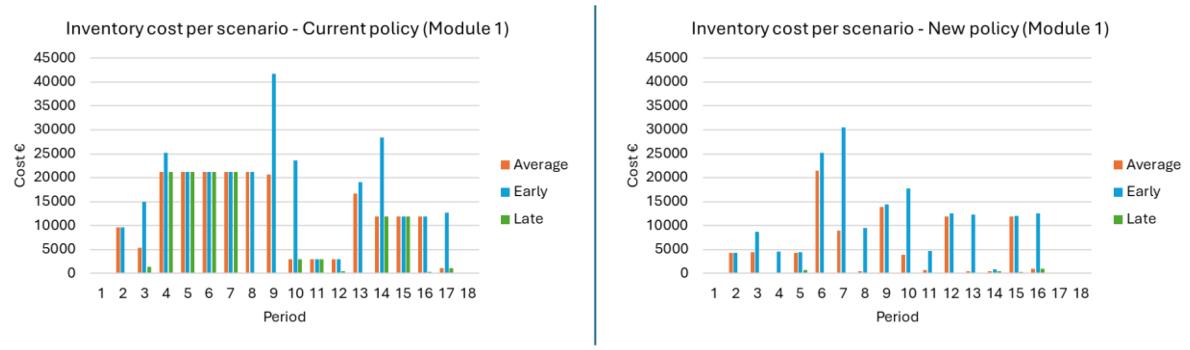


Figure 27: Inventory cost per scenario of the current policy and new policy for Module 1

5.4.1.3 Capacity usage

Figure 28 shows the capacity usage of the current and new setup policy respectively. In the current setup policy it can be seen that in period 2 and 4 the maximum capacity is exceeded and that in the other periods the used capacity is just below the maximum capacity. Considering the extension of the periods in the current policy, the actual current setup policy will probably exceed the maximum capacity in the other periods as well. In the new policy, it can be seen that around period 6 the used capacity is also high but does not exceed the maximum capacity. Since we allow setups to take place in multiple periods we can see that the capacity usage in the cleaning and unpacking area is manageable as the capacity usage is not consistently high.

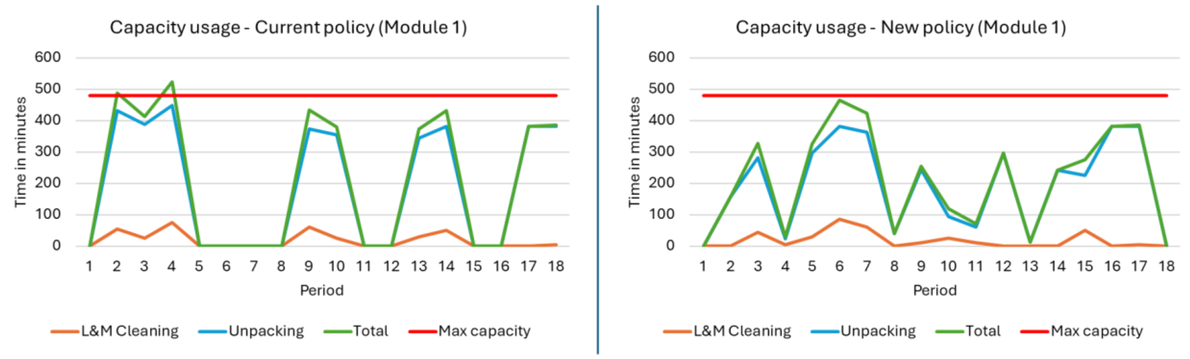


Figure 28: Capacity usage of the current policy and new policy for Module 1

5.4.2 Module 4

In the current situation, Systems-2 uses a setup policy for Module 4 in which there are four separate delivery moments: period 1, 2, 4, and 5. This setup policy is inserted in the MIP by fixing the setup

UNIVERSITY OF TWENTE.

variables for every item in the other periods to 0. This way the MIP is only allowed to make decisions based on the current setup policy.

5.4.2.1 Setup cost

Figure 29 shows the total setup cost per period of the current and new setup policy respectively. Only setups that consume capacity and therefore incur setup cost are relevant, as these are setups for materials that need to be delivered to the production area. The new setup policy makes use of more periods for the setups with the setups taking place in period 7. Here, similar to Module 1, the use of more periods for the setups is a consequence of taking the capacity into account which is not done in the current policy. However here, the total setup cost is €2134.37 for both the current and new policy. The reason for not having setups after period 7 is probably because in period 8 to 12 there is demand for the phantom items, which are assembled in the production area and do not require a setup. Module 4, includes relatively more phantom items compared to Module 1.

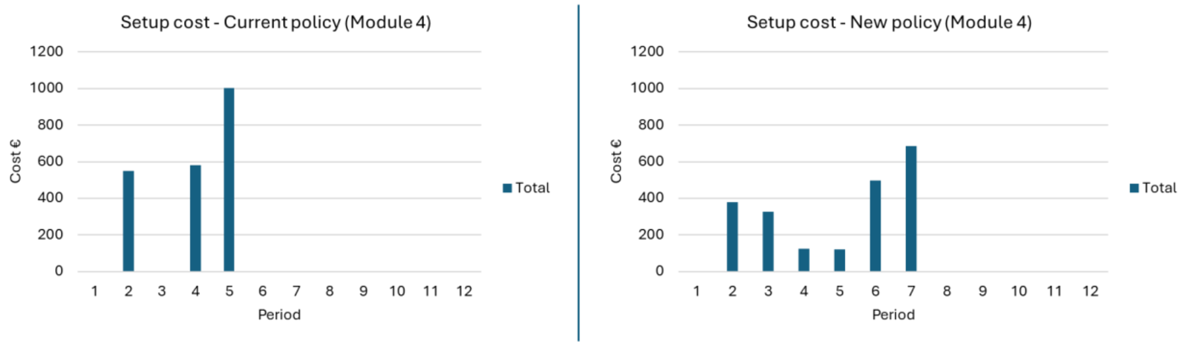


Figure 29: Setup plan of the current policy and the new setup policy for Module 4

5.4.2.2 Inventory cost

Figure 30 shows the inventory cost per scenario of the current and new setup policy respectively. Here it can be seen that the new setup policy impacts the inventory cost. The inventory costs are spread across multiple periods, resulting in less variation. Also, similar to Module 1, in the current policy there are inventory costs when items arrive late in period 2. Surely less than in Module 1, but it still illustrates that with such a policy items are delivered to the production area while they are yet needed. The new policy results in a total inventory cost of €5718.34 whereas the current policy has a total inventory cost of €5223.91. Here, similar to the setup cost, there are no inventory cost after period 7 in the new policy. This is caused by the fact that phantom items do not have inventory cost. These items are not purchased and therefore have no value leading to no holding cost.

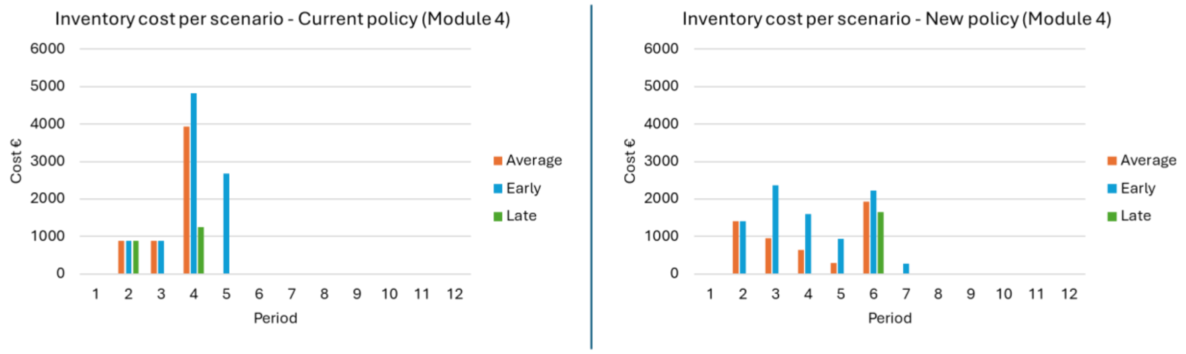


Figure 30: Inventory cost per scenario of the current policy and new policy for Module 4

5.4.2.3 Capacity usage

Figure 31 shows the capacity usage of the current and new setup policy respectively. With the current policy, the used capacity is equal to the maximum capacity in period 2 and exceeds the maximum capacity in period 4 and 5, while setups can take place in multiple other periods to balance the workload for the cleaning and unpacking area. With the new setup policy, only in period 7 the used capacity is equal to the maximum capacity and the workload in both the cleaning and unpacking area is more balanced.

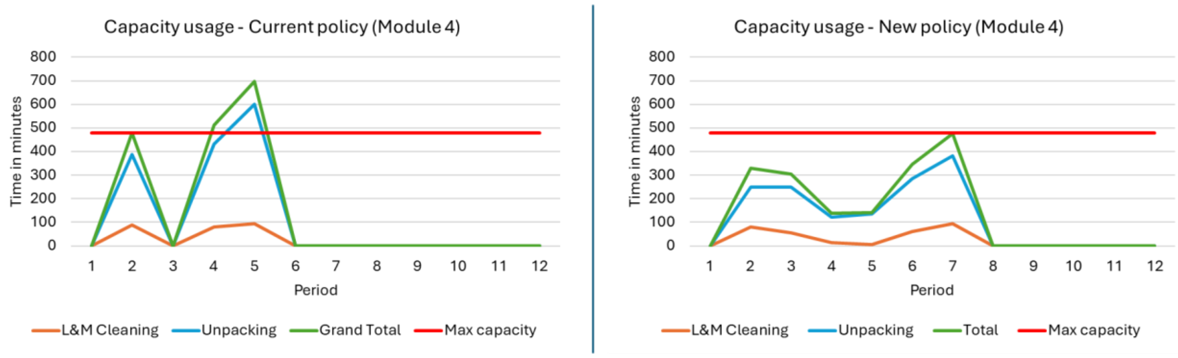


Figure 31: Capacity usage of the current policy and new policy for Module 4

5.5 Sensitivity analysis

In this paragraph, a sensitivity analysis is conducted to test the robustness and validity of the developed model. Two different aspects are analyzed: the impact of changing the maximum available capacity in the L&M cleaning area and unpacking area, and the impact of varying the number of periods. These analyses help to understand how sensitive the model's performance and outcomes are to changes in key input parameters, and to what extent the model remains effective under different conditions. For this sensitivity analysis we run the model on the input data of Module 4.

5.5.1 Capacity

In Section 4.2 the assumption is made that there is 1 FTE available for the cleaning and unpacking area. This results in a total available capacity of 480 minutes period, of which 96 minutes are allocated to the L&M cleaning area and 384 minutes to the unpacking area. Since the capacity constraint plays a critical role in how a setup plan is constructed it is relevant to investigate how changes in the maximum available capacity can affect the objective value. To explore this, we test two scenarios:

- A scenario with 0.75 FTE, resulting in a total available capacity of 360 minutes per period (72 minutes for the L&M cleaning area and 288 minutes for the unpacking area), and
- A scenario with 1.25 FTE, resulting in a total available capacity of 600 minutes per period (120 minutes for the L&M cleaning area and 480 minutes for the unpacking area).

We expect that a reduction in available capacity (0.75 FTE) will lead to a higher objective value, as the model has less flexibility to schedule setups due to a tighter capacity constraint. This may result in higher inventory cost or even overtime cost. Conversely, when there is more capacity available (1.25 FTE) we expect the objective value to decrease as the capacity constraint becomes looser allowing the model to have more flexibility in scheduling setups across the periods.

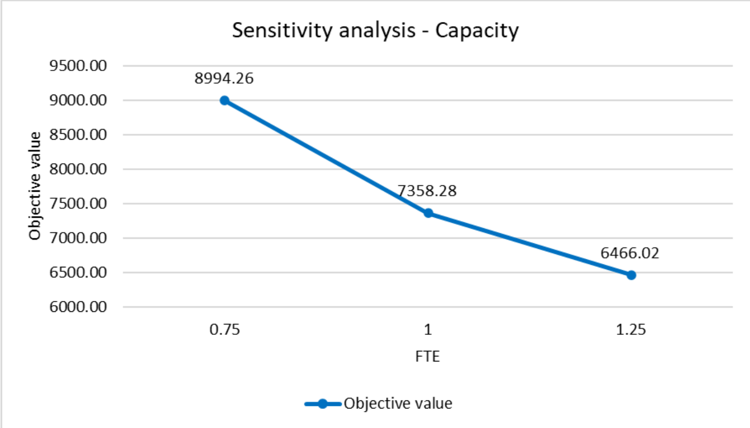


Figure 32: Sensitivity analysis on available capacity

Figure 32 shows that having a reduced available capacity (0.75 FTE) results in a higher objective value, and having an increased available capacity (1.25 FTE) results in a lower objective value. This confirms the expected relationship between the available capacity and the objective value, which supports the model's consistency. Appendix C shows the capacity usage for both scenarios.

5.5.2 Planning horizon

The motivation for this research, described in Section 1.2, is that the expected increase in the move rate will eventually lead to having Systems-2 to reduce the throughput time of each module. A reduction in the throughput time of a module would result in the reduction in the planning horizon for a module. Similar to the available capacity, the planning horizon plays a critical role in how a setup plan is constructed. However, unlike the capacity, it does not directly tighten or loosen a constraint. In this sensitivity analysis we investigate how changes in the number of periods will affect the objective value. To explore this, we again test two scenarios:

- A scenario with a move rate of 1.20, which results in a planning horizon of 13 periods, and
- A scenario with a move rate of 1.67, which results in a planning horizon of 10 periods.

Since the planning horizon does not directly constrain the model, it is difficult to predict a trend in the objective value. We expect however the model should find similar results to the original scenario with 12 periods.

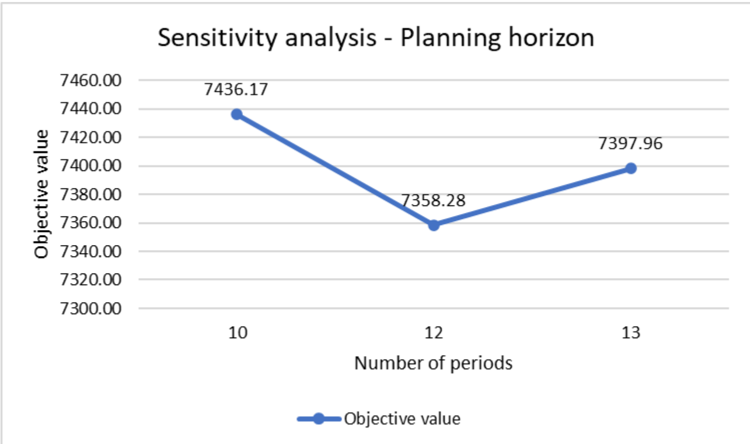


Figure 33: Sensitivity analysis on the planning horizon

Figure 33 shows that the scenario with a planning horizon of 13 periods finds an objective value that is close to the original scenario. The scenario with a planning horizon of 10 periods also produces a similar objective value, although the difference compared to the original is slightly larger. This could be caused by the fact that a change of two periods has more impact than a change of 1 period. The graph does show that the model is able to produce consistent results under varying planning horizons, which supports the models robustness. Appendix D shows the setup cost per period for both scenarios.

5.6 Summary

By experimenting with multiple configurations of the level of interrelatedness between setup variables, we have shown that a configuration with $l = 1$ for condition 2 and $l = 2$ for conditions 1 and 3 results in a best performance in terms of selecting setup variables to be destroyed in a solution, thereby creating a subproblem. This configuration is able to solve subproblems to optimality in 4.75 seconds while also finding improvements in the overall solution in most cases. In contrast, most of the configurations either reach the set time limit or fail to find improvements to the solution in most cases. The experiment regarding the local search and diversification strategy, showed that a random diversification strategy performs better compared to a diversification strategy based on logic. A local search strategy that randomly selects the first setup variable from a constructed solution showed the best performance. Although, a greedy approach based on setup cost showed similar results, a random approach proved to be more consistent and reliable.

By applying this configuration of the model to two separate modules, and inserting the current setup policy into the MIP model, we have been able to compare the new setup policy to the current setup policy. Both new setup policies for Module 1 and Module 4 incorporate additional delivery moments which differ from the current setup policies of both modules. The use of additional delivery moments helps to distribute the inventory cost more evenly over multiple periods, with the effect most noticeable at Module 4. The new setup policy is able to almost eliminate the issue of delivering materials prior to when they are needed in production. The use of additional delivery moments also helps to balance the workload in the cleaning and unpacking area, ensuring that the available capacity is not exceeded.

The sensitivity analysis both the impact of changing the maximum available capacity in the L&M cleaning area and unpacking area, and the impact of varying the number of periods showed results that were in line with the expectations. This supports that the model performs consistent and robust.

6 Implementation

This chapter describes how the model can be used and implemented by Systems-2. This is done by answering the following research question:

How can the model and solution be implemented and used by Systems-2?

Section 6.1 describes how the proposed solution method can be applied in practice. First we outline how the results of the model can be integrated into the planning process and used by end-users. In Section 6.2 we present the technical implementation, including the required inputs, software setup and how to run and interpret the model.

6.1 Functional implementation

In the problem description in Section 1.3, it is mentioned that Systems-2 wants to achieve a lower WIP inventory in the production, cleaning, and unpacking area. Four core problems are identified for the main problem:

1. Too many parts are delivered at once
2. A single delivery contains multiple load carriers
3. Parts are allocated in wrong sets
4. There is no alignment between reality and the planned schedule through an ERP-system

In the problem description, the four core problems were introduced as parallel contributors to the high WIP levels in the production, unpacking, and cleaning areas. However, based on the findings of this research, it becomes clear that core problem 1 and 2 can serve as a foundation for solving core problem 3 and 4. Rather than addressing all four problems independently, a more integrated approach can be adopted: by providing structured delivery quantities and timing to design more effective set allocation rules for core problem 3 and to better align the ERP system with the actual production flow for core problem 4. This section describes the required steps that Systems-2 must undertake to allocate parts in the right sets and to create an alignment between reality and the planned schedule through an ERP-system such that the WIP in the production, cleaning, and unpacking area can be minimized in the future

6.1.1 Allocation of parts into material sets

Currently, all materials are grouped into a set (processing step) and delivered to the production area at once. However, due to an incorrect allocation of materials to processing steps, materials are often supplied too early – before they are actually needed – which leads to idle inventory in the production area. This misallocation can also influence the WIP inventories at the cleaning and unpacking area. These materials should not be cleaned or unpacked at that moment and in cases where there is no capacity reserved or available it causes the WIP inventories to be higher.

To accurately form sets of materials that can be delivered at once to the production area, we must know the upper limits of how many materials can be handled in total and per load carrier. Our model provides a guideline for the maximum quantity to be delivered to the production area in each week (period). Instead of assigning all required materials to a few processing steps upfront, materials can now be split across multiple smaller processing steps based on their actual production need and the available capacity in the unpacking and cleaning area. By aligning the allocation of materials with the delivery quantities suggested by the model, materials are only released into the system when truly necessary.

In practice, this means that the Production Assistant – required for the material planning – can use the model output as a planning tool that is aware of the available capacity. The quantities suggested by the model serve as upper bounds when deciding how many parts to include in a set. The PA can then collaborate with a Factory Engineer of the module in question to form accurate material sets, as the Factory Engineer possesses required information about the materials that are being used in a module. This can prevent cleaning and unpacking before it is needed, ensures a smoother production flow, and contributes directly to reducing the WIP inventory levels.

6.1.2 Creating an alignment between reality and the ERP-system

Currently, when an order is placed by the customer, Systems-2 receives a delivery date from the customer, which is inserted in the ERP-system, BAAN. BAAN then calculates when production should start and spreads the processing steps across the production timeline. In practice it might occur that a certain milestone is not met. This however is not communicated back to BAAN making the production plan redundant. As a result, Systems-2 relies on verbal communication to plan the production, leading to potential errors, which often results in high WIP inventories at the production, cleaning and unpacking area.

At the point of writing, VDL ETGA is transitioning to a new ERP-system called Infor LN which is expected to be in use in 2025. This transition in combination with a reallocation of material sets makes it hard to setup a detailed plan for creating an alignment between reality and the new ERP-system. However, with the model's output that serves as a capacity aware planning tool and accurate material sets, it is now possible to provide meaningful input to the ERP-system. The system can incorporate updated processing steps with known capacity usage and the maximum available capacity per week for the cleaning and unpacking areas. This data allows the ERP-system to adjust and update production plans dynamically.

From a more practical perspective, Systems-2 must look at the possibilities of Infor-LN regarding the material and production planning. For instance, real-time signals, such as a notification for completing a processing step must be incorporated to continuously align the production schedule with actual progress. This enables a more accurate and resilient production planning, reducing the need for verbal communication and ad-hoc decision making that could lead to potential errors in the production planning.

6.2 Technical implementation

This research is limited to Module 1 and Module 4, therefore it is up to Systems-2 to run the model for Module 2, Module 3A and Module 3B. This section describes the necessary steps for putting the model to use. First, input data must be prepared for every module. This starts by exporting the bill-of-materials of a module from BAAN to Excel. Once this is done, the BOM must be filtered on the following two warehouse locations:

605: Purchase items
625: Phantom items

From this selection, all the items that have quantity of 0 or a unit in weight or litres must be removed. This leaves all the relevant items for our model. Second we create a worksheet for the sets that we use in our MIP (Items, Parent items, Phantom items, periods, scenarios, and Areas). Then, create a separate worksheet for the Gozinto, inventory cost, setup cost, setup time, end item demand, lead time, and capacity. The creation of these separate worksheets is trivial and should be achieved easily since the structure can be copied from the input data of Module 1 and Module 4 and only the items and their data has to be replaced by the items and the data of the module in question.

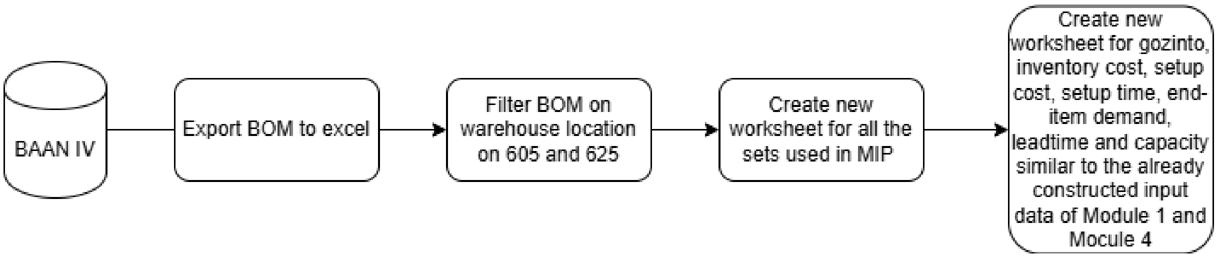


Figure 34: Overview of input data preparation for remaining modules

Now that the input data can be established, we must setup the environment in which the model and algorithm runs. The optimization model is written in python and developed using Visual Studio Code. To execute the model and interactively analyse the results during the development of the model, Jupiter Notebook is used. Thus, in order to run the model Systems-2 must install the Jupyter extension and the python extension on Visual Studio Code. Installing both extensions allows VS code to support the use of Jupyter and Python.

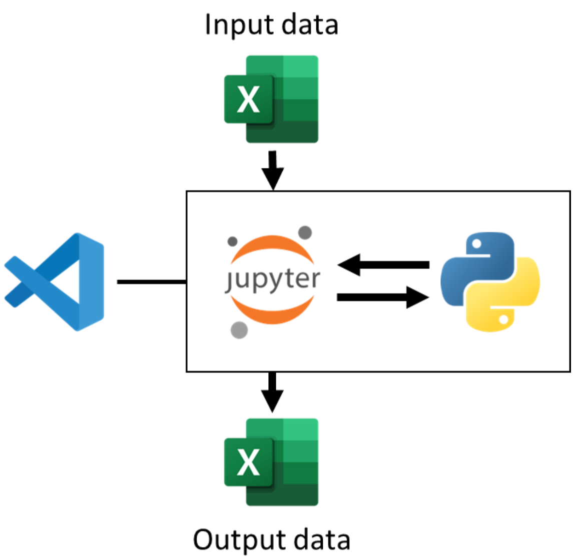


Figure 35: Model environment

However in order to actually execute Python code, the latest version of Python must be downloaded and installed. Once these steps are followed and the environment is set up, the model can be put to use by Systems-2.

6.3 Summary

The developed model provides information on the quantity to be ordered in a period such that there are no capacity issues in the cleaning and unpacking area and the inventory costs are minimized. Instead of assigning all required materials to a few processing steps upfront, materials can now be split across multiple smaller processing steps can be formed by the PA and a factory engineer, based on the actual production need and the available capacity in the unpacking and cleaning area. Once Infor LN is put to use, Systems-2 must investigate the possibilities for creating an alignment between the production planning and actual status of the production.

The model has been applied to two of the in total five modules. For Systems-2 to be able to apply the model to the remaining three modules, first the data must be prepared for each module. And second, an environment must be created in which the model can be executed.

7 Conclusion and further research

This chapter presents the conclusions of this research along with its limitations and suggestions for further research. Section 7.1 we answer the main research question. Section 7.2 describes the limitations of this research followed by recommendations for further research. Section 7.3 describes the theoretical and practical contribution of this research.

7.1 Conclusion

Currently, Systems-2 experiences that there is not sufficient space in the production area to achieve a higher move-rate for the production of their modules. Part of this problem is caused by the fact that the WIP inventory in the production, cleaning and unpacking area is too high. To address this challenge the following main research question was formulated:

How can the WIP inventory at the production, cleaning and unpacking area be controlled while taking maximum capacity at the cleaning and unpacking area, and different types of load carriers into account?

Answering the sub-research questions has allowed us to develop a MIP model based on the multi-level capacitated lot sizing problem. Applying this model to two modules demonstrated that a new setup policy that makes use of additional delivery moments can control the WIP inventory at the production, cleaning and unpacking area. By using additional delivery moments the inventory cost and capacity are distributed more evenly across the periods. The new setup policies were able to reduce the inventory cost for both modules with a significant reduction in inventory cost of 56,6% for Module 1 while never exceeding the available capacity for both the cleaning and unpacking area.

This research also demonstrates the impact of different configurations within the proposed solution approach. The solution method is based on a destroy-and-repair mechanism, where the strategy for selecting which setup variables to destroy plays a critical role in both the computational efficiency and the quality of the solution. An experiment that tested different combinations of local search and diversification strategies showed that selecting the first setup variable for the destroy strategy can best be done randomly for local search and diversifying to a different neighborhood. An experiment for selecting additional setup variables to destroy showed that a configuration for the level of interrelatedness that focuses on selecting setup variables that are interrelated through periods and capacity usage performs best. This configuration solves subproblems to optimality in 4.75 seconds and finds improvements in the overall solution in most cases, unlike other configurations that either hit the time limit or failed to improve the solution consistently.

7.2 Limitations and further research

A setup policy is largely determined by the timing of demand for each item. In our model demand is triggered once the parent item is required in production. Although this structure captures the dependencies between items, it assumes immediate availability of predecessor items once their parent is needed. In practice, it takes time to assemble these items, this assembly time is not incorporated in the model due to a lack of accurate data on assembly times.

Furthermore, the motivation for this research describes that there is not enough space in the production, cleaning and unpacking area. This research addresses this by minimizing the inventory cost which indirectly reduces the inventory. However, it does not account for the physical dimensions of items. For instance, it can be the case that an item with low inventory cost use a lot of space due to its size. It would be better to have this item just in time rather than a small item that has a higher inventory cost. However, measuring the dimensions of all the items that are delivered to the production area is too time-consuming.

In addition, assumptions were made for the setup times of crates, piping, and pallets that eventually determine the setup cost. To improve the model accuracy, actual setup times could be recorded in practice, allowing for a more precise cost estimation.

In the numerical study, an experiment is conducted that investigates what combination of local search and diversification strategy results in the best performance. The experiment showed that a (Random / Random) combination provided the best results over 10 runs. However, the combination of (Setup cost / Random) provided similar results, but proved to be a little less consistent. It is therefore interesting to conduct an additional larger-scale experiment that compares both combinations to perform a statistical analysis and draw more reliable conclusions.

As mentioned, this research limits to two of five modules. In Chapter 6 we proposed that the model should be applied to the remaining modules. All modules have a dedicated workspace and can therefore be studied separately, but they all make use of the same delivery processes. Therefore it could be interesting to see what the effect is on the cleaning and unpacking area if we do not treat each module separately.

The deliverable of this research is an MIP model. The output of this model is now exported to an excel file. To make the output of the model better available for the employees at Systems-2, an application must be made that presents the output of the model in such a way that it is easy to understand and analyze.

7.3 Theoretical and practical contribution

In Chapter 3 we identified comparable studies during the literature study that primarily focused on developing new solution approaches to be able to find quality solutions for a generic version of the MLCLSP. These new solution approaches were all tested on predefined data sets in order to make accurate comparisons. Chapter 2 showed that every module consist of items that can fit into different categories. It also showed that our setup is not related to a machine but to the two separate delivery processes. This created a challenge in adapting the generic MLCSLP to a problem specific MLCLSP model for Systems-2 capable of handling real-life data inputs. The model therefore also contributes to the theoretical understanding of material and production planning processes, particularly in environments where lot-sizing decisions play a central role.

The primary practical contribution of this model is that it serves as a foundation for redesigning the material sets that must be delivered to the production area. With the output data of the model, Systems-2 is now able to design material sets such that the WIP inventory in the production, cleaning and unpacking area is controlled. An additional practical contribution is that this model can be applied to the remaining three modules. This enables Systems-2 remove idle inventory at every workspace to create more space for the production of the modules in order to achieve a higher move-rate.

Bibliography

Arriola, L., & Hyman, J. M. (2009). Sensitivity Analysis for Uncertainty Quantificaiton in Mathematical Models. *Mathematical and Statisctical Estimation Approaches in Epidemiology*, 195-247.

Bahl, H. C., Ritzman, L. P., & Gupta, J. N. (1987). Determining lot sizes and resource requirements: A review. *Operations Research*, 329-345.

Billington, P. J., McClain, J. O., & Thomas, L. J. (1983). Mathematical programming approaches to capacity-constrained MRP systems: Revie, Formulation and problem reduction. *Management Science*, 1126-1141.

Bruno, G., Genovese, A., & Piccolo, C. (2014). The capacitated lot sizing model: A powerful tool for logistics decision making. *Int. J. Production Economics*, 380-390.

Buschkühl, L., Sahling, F., Helber, S., & Tempelmeier, H. (2008). Dynamic capacitated lot-sizing problems: a classification and review of solution approaches. *OR Spectrum*, 231-261.

Chen, H. (2015). Fix-and-optimize and variable neighborhood search approaches for mmulti-level capacitated lot sizing problems. *The International Journal of Management Science*, 25-36.

Chen, W.-H., & Thizy, J.-M. (1990). Analysis of relaxations for the multi-item capacitated lot-sizing problem. *Annals of Operations Research*, 29-72.

Clark, A. R., & Armentano, V. A. (1995). The application of valid inequalities to the multi-stage lot-sizing problem. *Computers & Operations Research*, 669-680.

Comelli, M., Gourgand, M., & Lemoine, D. (2008). A review of tactical planning models. *Journal of Systems Science and Systems Engineering*, 204-229.

Emelogu, A., Chowdhury, S., Marufuzzaman, M., Bian, L., & Eksioglu, B. (2016). An enhanced sample average approximation method for stochastic optimization. *International Journal of Production Economics*, 230-252.

Glock, C. H., & Grosse, E. H. (2014). The lot sizing problem: A tertiary study. *Int. J. Production Economics*, 39-51.

Heerkens, H., & van Winden, A. (2021). Action problems. In H. Heerkens, & A. van Winden, *Solving Manegerial Problems Systematically* (p. 22). Groningen: Noordhoff Uitgevers bv.

Helber, S., & Sahling, F. (2010). A fix-and-optimize approach for the multi-level capacitated lot sizing problem. *Int. J. Production Economics*, 247-256.

Hu, Z., & Hu, G. (2016). A two-stage stochastic programming model for lot-sizing and scheduling under uncertainty. *International Journal of Production Economics*, 198-207.

Jans, R., & Degraeve, Z. (2006). Modeling industrial lot sizing problems: A review. *International Journal of Production Research*, 1619-1643.

Jans, R., & Degraeve, Z. (2007). Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches. *European Journal of Operational Research*, 1855-1875.

Jaruphongsa, W., Çetinkaya, S., & Lee, C.-Y. (2007). A dynamic lot-sizing model with multi-mode replenishments: polybomial algorithms for special cases with dual and multiple modes. *IIE Transactions*, 453-467.

Jeunet, J., & Jonard, N. (2000). Measuring the perfomance of lot-sizing techniques in uncertain environments. *International Journal of Production Economics*, 197-208.

Karimi, B., Fatemi Ghomi, S., & Wilson, J. (2003). The capacitated lot sizing problem: a review of models and algorithms`. *The International Journal of Management Science*, 365-378.

Rardin, R. L., & Uzsoy, R. (2001). Experimental Evaluation of Heuristic Optimization Algorithms: A tutorial. *Journal of Heuristics*, 261-304.

Schlenkrich, M., & Parragh, S. N. (2024). Capacitated multi-item multi-echelon lot sizing with setup carry-over under uncertain demand. *International Journal of Production Economics*.

Tempelmeier, H., & Derstroff, M. (1996). A lagrangian-based heuristic for dynamic multi-level multi-item constrained lotsizing with setup times. *Management Science*, 738-758.
VDL Groep. (2024, September 13). *Over VDL Groep*. Retrieved from VDL Groep: https://www.vdlgroep.com/nl
Verweij, B., Ahmed, S., Kleywegt, A. J., Nemhauser, G., & Shapiro, A. (2003). The Sample Average Approximation Method Applied to Stochastic Routing Problems: A Computational Study. *Computational Optimization and Applications*, 289-333.
Wagner, H. M., & Whitin, T. M. (1958). Dynamic version of the economic lot size model. *Management Science*, 89-96.

Appendix A

Algorithm 3. Fix-and-optimize $(\bar{Y}, l, r_k)$

Solve the linear relaxation of model $MLCLSP(\bar{Y}, r_k)$ *to get* $\hat{Y} = \{\hat{Y}_{it}\}$ *and* $\boldsymbol{\Omega} = \{ Y_{it} \,|\hat{Y}_{it} \neq \bar{Y}_{it}\}$*;*
$iter \leftarrow 0$;
Repeat
    $iter \leftarrow iter + 1$;
    Randomly choose a pair Y_{it} *from* $\boldsymbol{\Omega}$ *with the same probability for each element in* $\boldsymbol{\Omega}$*;*
    Solve subproblem $SP^l_{i,t}(+\boldsymbol{\Omega})$*;*
    If *the solution of* $SP^l_{i,t}(+\boldsymbol{\Omega})$ *is better than the current solution of model MLCLSP,* ***Then***
        Set $\bar{Y}$ *to the setup plan of solution* $SP^l_{i,t}(+\boldsymbol{\Omega})$*;*
        Solve the linear relaxation of model $MLCLSP(\bar{Y}, r_k)$ *to get* $\hat{Y} = \{\hat{Y}_{it}\}$ *and* $\boldsymbol{\Omega} = \{ Y_{it} \,|\hat{Y}_{it} \neq \bar{Y}_{it}\}$*;*
        $iter \leftarrow 0$;
    End If
Until $iter \geq |\, \boldsymbol{\Omega} \,|$*;*

Algorithm 4. Swap-fix-and-optimize $(\bar{Y}, l, r_k)$

$TL = \emptyset$*;*
For $iter = 1 \; to \; k$ ***do***
    Solve the linear relaxation of model $MLCLSP(\bar{Y}, r_k)$ *to get* ***Ω;***
    Swap = False;
    While *Swap = False*
        Randomly choose a pair $Y_{i_r t_r} \in \boldsymbol{\Psi}\backslash \mathrm{TL}$ *with the same probability for each element in* $\boldsymbol{\Psi}\backslash\mathrm{TL}$*;*
        Swap the value of binary variable $Y_{i_r t_r}$ *from* $Y_{i_r t_r} = \bar{Y}_{i_r t_r}$ *to* $Y_{i_r t_r} = 1 - \bar{Y}_{i_r t_r}$*;*
        Solve the subproblem of model MLCLSP that reoptimizes all setup variables in $IR^l(Y_{i_r t_r}, +\boldsymbol{\Omega})\backslash\{TL \cup \{Y_{i_r t_r}\}\}$*, while fixing all setup variable in* $FB^l(Y_{i_r t_r}, -\boldsymbol{\Omega}) \cup \{TL \cup \{Y_{i_r t_r}\}\}$*, where* $Y_{i_r t_r}$ *is fixed to* $1 - \bar{Y}_{i_r t_r}$ *and all other variables* Y_{it} *in* $FB^l(Y_{i_r t_r}, -\boldsymbol{\Omega}) \cup TL$ *are fixed to* $\bar{Y}_{i_r t_r}$*;*
        If *the solution of the subproblem is acceptable,* ***Then***
            Update $\bar{Y}$ *by the setup plan of the solution;*
            $TL \leftarrow TL \cup \{Y_{i_r t_r}\}$*;*
            Swap = True;
        End If
    End while
End for

Appendix B

Algorithm 5. Solution approach for improving MLCLSP solution

Solve the relaxation of the MLCLSP for 60 seconds to get a relaxed solution;
Set $y_{it} = 1$ for every value of $y_{it} > 0$ in the relaxed solution, 0 otherwise;
Insert the rounded values of y_{it} into the MIP to generate a feasible initial solution;

Set local optimum to the objective value of the initial solution;
Set local setup plan to initial setup plan;
Set current best solution to the objective value of the initial solution;

VNS = False;
Greedy = True;

Set time limit;
Initialize start time;

While computation time < time limit:
    Iteration = 0;
    n = 5;

    While iteration < n:
        iteration = iteration + 1

        If VNS = True:
            Select first pair (i, t) based on selected diversification strategy (except for pairs in tabu list)
        Else:
            Select first pair (i, t) based on selected local search strategy

        If Greedy = True
            Fixed_setup = local setup plan;
        Else:
            Fixed_setup = current best plan;

        Select pairs (i, t) that are interrelated through the set conditions to form IR^1
        For every pair $(\boldsymbol{i}, \boldsymbol{t})$ in $\boldsymbol{IR^1}$:
            Select pairs (i, t) that are interrelated through the set conditions to form IR^2

        For every pair $(\boldsymbol{i}, \boldsymbol{t})$ in $\boldsymbol{IR^2}$:
            Remove every value of y_{it} from fixed setup plan;

        Insert fixed_setup in MIP to find new objective and new setup plan;
        VNS = False;
        Greedy = True;
        If new objective < local optimum:
            Local optimum = new objective;
            Local setup plan = new setup plan;
            iteration = 0
        End If

Store first selected pair (i, t) *in tabu list;*
If length of tabu list > 10:
Remove first entered pair (i, t)*;*
End If

VNS = True;
Greedy = False;
If local optimum < current best solution:
Current best solution = local optimum;
Current best plan = local setup plan;

Re-initialize local optimum to a large enough number;

Appendix C

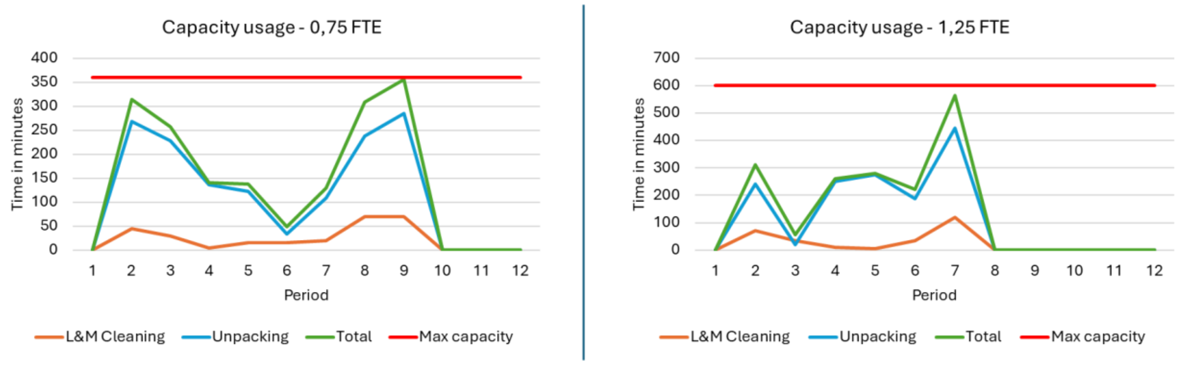
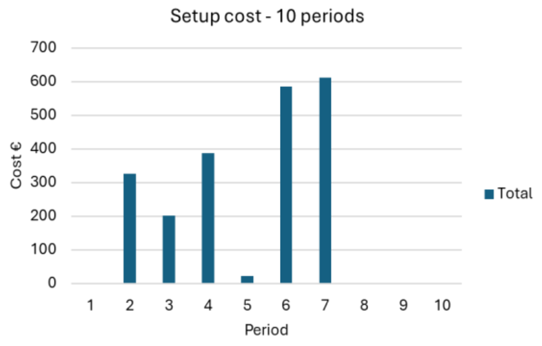


Figure 36: Capacity usage with 0.75 FTE and 1.25 FTE

Appendix D



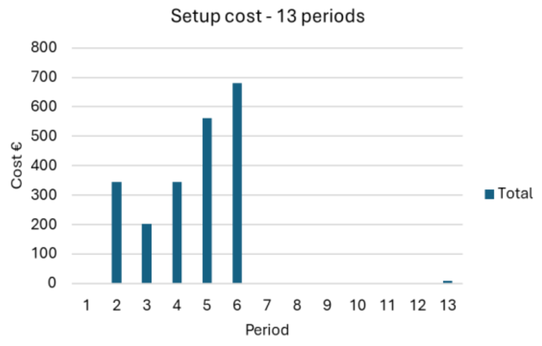


Figure 37: Setup plan with 10 periods and 13 periods