Throughput time improvement at Trioliet

BSc Industrial Engineering and Management

6/13/2024

Yur van Helden (Student B-IEM)

University of Twente supervisors
dr.ir. J.M.J. Schutten (Marco)
dr. D.R.J. Prak (Dennis)

Trioliet supervisor
Nijmeijer (Sjoerd)

Educational institution
University of Twente
Drienerlolaan 5
7522 NB Enschede

Company
Trioliet
Kleibultweg 59
7575 BW Oldenzaal

Preface

Dear reader,

The thesis is titled "Throughput time improvement at Trioliet" and it serves as the final assignment for my Bachelor of Industrial Engineering & Management. I would like to thank multiple groups of people who helped me during this process.

First, I thank the company for providing the opportunity to conduct this research. Especially, I am thankful to Sjoerd and Casper for their helpful guidance, support and patience. Additionally, I would like to thank all my colleagues from Trioliet, who always created a friendly atmosphere.

Secondly, I would like to thank my first supervisor, Marco, for always providing helpful and critical feedback and assisting me to improve the quality of my report. Further, I would also like to thank my second supervisor, Dennis, for providing clear and critical feedback.

Finally, I would like to thank my family roommates and my peers at the University of Twente (Camiel, Iris and Josefien) for supporting me during this period.

Kind regards,

Yur van Helden
June 2024

Management summary

Introduction

Trioliet is an agricultural company specialized in the development, design, and production of feed mixers, feeding systems, and silage cutters. This research focuses on the production line of the silage cutters, also known as the TU line. Currently, the TU line is experiencing high throughput times, which the company aims to reduce. The action problem, core problem, and main research question are formulated as follows:

- Action problem: The average throughput time per month at the TU line is too high.
- Core problem: The workload between workstations is not balanced.
- Main research question: How can Trioliet reduce the average throughput time by solving the workload imbalance at the TU line?

Current situation

The TU line assembles four product variants of their silage cutters, called Turbobuster (TU): a standard variant of standard TUs, a standard variant of TU180-XL, a configured variant of standard TUs and a configured variant of TU180-XL. The line consists of 8 workstations and each station has a set of tasks, depending on the product variant.

The workload distribution of the TU line in 2022 is determined with the processing times of tasks and product demand from the year 2022. Process times are collected through informal interviews with line operators and product demand is based on Excel sales records. Table 1 shows the current average workload distribution (per month) in the year 2022. The distribution shows that workstations 2 and 8 have the highest average workload per month. Workstation 2 performs many essential tasks at the beginning of the process and workstation 8 assembles, mounts and tests many ‘options’ (attachments with additional features) for different product variants. These workstations are the bottlenecks of the production line because they have the highest average workload and their workforce capacity has not been adjusted to accommodate the higher average workload (workstations 2 and 8 are utilized by one operator). From the bottleneck analysis, we conclude that we want to reduce the workload of workstations 2 and 8. We achieve this by applying changes to the task distribution of the TU line.

Workstations	1	2	3	4	5	6	7	8
Average workload (hours per month)	13.67	81.25	32.16	26.00	35.67	30.67	24.33	49.07

Table 1: Current workload distribution at the TU line (per month in 2022)

Solution approach

In literature, balancing problems for production lines include either rearranging tasks at an existing production line or assigning tasks to an empty production line. Our approach involves two steps: first, creating initial solutions and then improving these initial solutions.

The first step is to create initial solutions. We already have the current task distribution and we generate three additional initial solutions using a constructive heuristic, where we select multiple tasks based on the priority rule 'precedence position' and then we choose one task randomly from the set of selected tasks. The selected task is assigned to the first available workstation according to the precedence and capacity constraints of a workstation.

The second step is to improve the initial solutions with an improvement heuristic. We use ‘hill climbing’, where tasks are swapped or moved greedily from the highest and second-highest workstation to workstations with a lower workload. We use two neighborhoods: move (nh1) and swap (nh2). The first improvement of move is selected for nh1 and the first improvement of swap is selected for nh2. The

best operator out of both neighborhoods is accepted (nh1, nh2). The Mean Absolute Deviation (MAD) is used to assess 'workload balance'.

Results

The results of the four solutions generated, by applying a constructive and hill climbing heuristic, show that they all improve the workload balance compared to the current situation. We select the best solution based on the performance measures for 'workload balance', using four objectives: 'Mean Absolute Deviation' (MAD), 'minimum workload', 'maximum workload' and 'difference between maximum and minimum workload'. We conclude that solution 3 is the best solution for Trioliet's problem, called proposed solution, because it performs best on objectives 'min' and 'diff min-max' (Table 3). The 'MAD' value ended up at 9.85 for all solutions because it ended up in a local optimum. The 'maximum workload' value ended up 76 hours for all solutions, because at a certain point, no more tasks could be removed from the bottleneck workstation.

	Current situation	Solution 1	Solution 2	Solution 3	Solution 4
MAD	14.27	9.85	9.85	9.85	9.85
Min	13.67	19.67	22.49	28.50	18.99
Max	81.25	76.00	76.00	76.00	76.00
Diff min-max	67.58	56.33	53.50	47.50	57.01

Table 2: Objective values "workload balance" of alternative solutions and the current situation

Then, the task and workload distribution of the proposed solution was compared with the current situation at the TU line. The workload distribution has changed, where the workload of workstations 2 and 8 have decreased and the other workstations have remained in balance. Figure 1 shows the workload distribution of the current situation and the proposed solution. In total, we have reallocated 18 tasks compared to the current situation. Four conclusions can be drawn:

1. Tasks related to independent assemblies of options and other parts were moved. For example, the assembly of the control block and multiple options.
2. Tasks related to the mounting of options and parts on the product were moved towards earlier workstations of the production process. For example, the valve and control block can be mounted to the product earlier (without attaching hoses).
3. The task 'hoses for options' is fully carried out at the last workstation due to the precedence relations.
4. The task 'testing of options' is fully carried out at the last station because it is the last task in the precedence diagram.

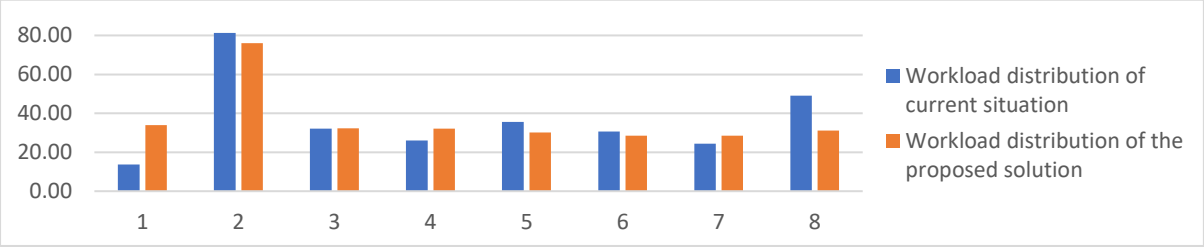


Figure 1: Workload distribution of current situation and the proposed solution

In the last part of the results, we translate the proposed workload balancing solution back into throughput time reduction. We estimate the percentual reduction in throughput time of the proposed solution by calculating throughput time of the production line, cycle time at workstations and (expected) waiting time at workstations. Throughput time is the sum of cycle time of all workstations. Cycle time is the sum of workload and waiting time at a workstation. The Kingman relationship is used to calculate the expected waiting time at a workstation, where expected waiting time equals the multiplication of variability, utilization, and lead time at a workstation. We assume that variability = 1, which means that the variability of a workstation is low. Further, we assume that variability remains constant when the task

distribution changes. Thus, in our research, the expected waiting time at a workstation is only determined by the utilization and lead time of a workstation.

The research idea is to reduce the overall waiting time of the TU line by significantly decreasing the waiting time at highly utilized workstations and smally increasing the waiting time at low utilized workstations. However, we do not know the utilization of workstations at the TU line because the number of operators at the TU line varies, and the distribution of line operators at the TU line is not known. Therefore, we conduct a sensitivity analysis to determine the effect of various values of the independent variable "workstation utilization" on the dependent variable "expected waiting time at a workstation ". We vary the workstation utilization of workstation 2 between 81 to 90% and assume an equal distribution of available capacity across workstations, so that the utilization of other workstations can be easily calculated. To conclude, the proposed solution resulted in a throughput time reduction ranging from 14.56 to 29.21% (Figure 2).

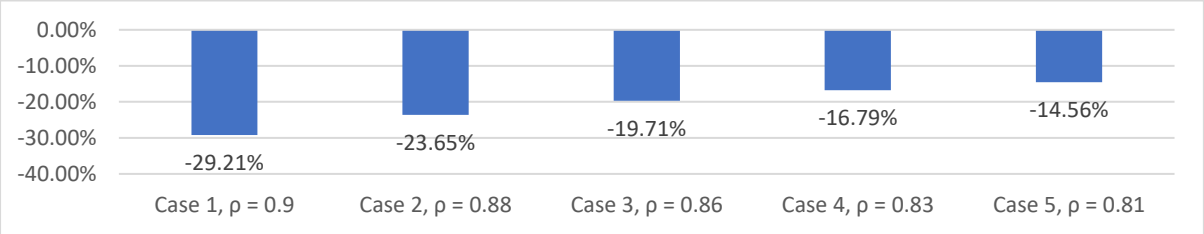


Figure 2: Throughput time reduction for the ρ cases for the proposed solution (sol 3)

Conclusion

In the conclusion of the research, we look at the feasibility of the solution and whether it is worthwhile to implement the solution.

The proposed solution has one issue regarding implementation, which is that the TU line has limited available space, which means the inventory, materials, worktables cannot be relocated.

Further, two points are stated to argue whether the solution is worthwhile to implement.

1. The proposed solution distributes complex tasks, like assemblies of options, across workstations, instead of having these tasks at one workstation (in the current situation). This means that multiple line operators should be able to execute the complex tasks, which is not the case now, thus more training is required to execute the proposed solution.

2. It is difficult to assess a precise impact as the result gave a range of potential throughput time improvements from 14 to 29%. The solution seems worthwhile to implement as the throughput time can be reduced by at least 14%.

In conclusion, the proposed solution seems worthwhile to implement, because the throughput time can be reduced by at least 14%. However, we recommend that the company evaluates the feasibility of the new task distribution of the proposed solution, whether available space is an issue. Additionally, the company should determine whether it is worth training the line operators so that they can perform complex tasks.

Contents

Glossary of terms

Definitions

Term	Definition
Turbobuster	The marketing name of the silage cutter that Trioliet assembles (Trioliet engineer)
TU line	The name of the production line of the Turbobuster (Trioliet engineer)
Throughput time	The time from the start of the first to the completion of the final processing step (also known as makespan) (Niemueller, 2017)
Workload (at a workstation)	The sum of the processing times of the tasks assigned to the workstation (Dinler, 2021)
Process time (of a task)	The duration of executing a task, also called task time.
Bottleneck	The most heavily loaded workstation (Pastor, 2021)
Assembly line	A flow-oriented production system where the productive units perform operations, stations are aligned in a serial manner (Boysen, 2006)
Assembly line balancing	Arranging individual processing and assembly tasks at the workstations so that the total time required at each workstation is approximately the same (Kiran, 2019)
Heuristic	Problem specific approach that employs a practical method that often provides sufficient accuracy for the immediate goals (Lindfield, 2019)
Local search heuristic	A method that iteratively improves a single solution, called current solution, by making small changes to it. (Glover, 2015)
Lead time	The average processing time in the workstation or the assembly line (J.Hopp, 2008)
Cycle time	Cycle time is made out of the following components: process time, queue time move time, setup time, wait-to-batch time, wait-in-batch time and wait-to-match time (J.Hopp, 2008)
Queue time (waiting time)	The time spent waiting for processing at the workstation or moved to the next workstation (J.Hopp, 2008)
Sensitivity analysis	Sensitivity analysis is an analysis that determines how different values of an independent variable affect a particular dependent variable under a given set of assumptions. (Kenton,2023).

Abbreviations

Abbreviation	Term
TU	Turbobuster
HPU	Hydraulic push-off unit
MPU	Mechanical push-off unit
TF	Turbofeeder
ES	Electrical system
ALB	Assembly line balancing
ALBP	Assembly line balancing problem
SALBP	Simple assembly line balancing problem
GALBP	General assembly line balancing problem
SUDLBP	U-shaped disassembly line balancing problem

MALBP	Multi-manned Assembly Line Balancing Problem
MIP	Mixed integer programming
TOTSIS	Multi-attribute decision-making approach
LCR	Largest Candidate Rule
K&W	Killbridge and Wester (heuristic)
RPWM	Rank Positional Weight Method
NFH	Number of Followers Heuristic
SA	Simulated Annealing
TS	Tabu Search
ILS	Iterative Local Search
HC	Hill Climbing
SD	Steepest Descent
MAD	Mean Absolute Deviation

Table of Figures

Table of Tables

1. Introduction

Chapter 1 introduces the research conducted for Trioliet. Section 1.1 provides a description of the company and Section 1.2 motivates the research and gives a short description of the production line at Trioliet. Next, Section 1.3 describes the problem with a problem cluster and identifies the core problem of the research. Section 1.4 states the research objective and scope. Section 1.5 describes the research limitations. Last, Section 1.6 explains the structure of the research with research questions.

1.1. Company description

Trioliet is a family-owned company that specializes in designing, producing and marketing livestock feeding machines. The company produces and sells multiple products for livestock farms: feed mixers, silage feeder, silage grab, silage cutter, feed management systems, weighing systems and many accessories. Additionally, Trioliet sells technical support and spare parts to customers.

Trioliet's products are sold through an international dealers' network, in countries like US, Germany and China, and it has become the market leader in The Netherlands. The company has grown over the years due to several reasons: a great understanding of the market, high product expertise and the company has distinguished itself for producing customer-specific machines. As a result, the company currently has around 300 employees and it makes an annual revenue of 100 million euros.

The company 'Trioliet' was established in 1950 by the Liet brothers in Purmerend and changed its name to 'Trioliet Mullos', after acquiring the company 'Mullos' in 1958. In 1997, the company headquarters and manufacture plants moved to Oldenzaal, The Netherlands, and it is still based there. This research takes place at the manufacturing plant in Oldenzaal, where many production lines are stationed.

1.2. Research motivation & description TU line

Trioliet is currently experiencing problems at the production line for their silage cutters. Silage cutter is referred to as 'Turbobuster' and the production line is called 'TU line'. Currently, the TU line is experiencing a high throughput time, which is defined as 'the time from the start of the first to the completion of the final processing step (also known as makespan)' (Niemueller, 2017). We formulate the action problem as 'the average throughput time per month at the TU line is too high'.

To provide context, the TU line consists of 8 workstations, where the Turbobuster is gradually built, and the entire process is manually operated with 3-to-5 operators.

1.3. Problem cluster

The problem is described with a problem cluster. Figure 3 depicts the problem cluster with causes and effects (Arrows point from cause to effect). The chosen core problem satisfies the criteria of Hans Heerkens (2016): the problem occurs, has relevancy, is influenceable and does not have its own causes.

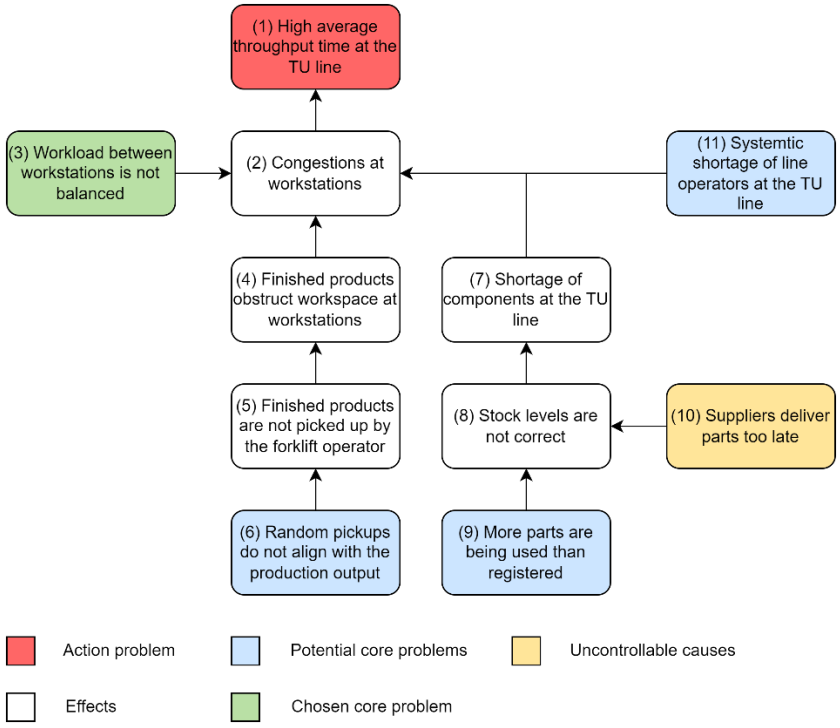


Figure 3: Problem cluster

The problem cluster starts with the action problem 'the average throughput time (per month) at the TU line is too high' (1) it is caused by congestion at workstations (2) which means that workstations slow down or stop the production process. There are 4 causes for congestions at the production line (2): workload between workstations is not balanced (3), finished products obstruct workspace at the production line (4), shortage of components at the production line (7) and systematic shortage of workers at the production line (11).

The TU line encounters workload imbalance between workstations (3), which means that workloads are not equally distributed among the workstations, whereby tasks are not properly distributed among workstations. Workload imbalance occurs because some workstations have significantly higher workloads, due to having more time-consuming tasks, compared to the other workstations. According to the Trioliet engineer, workstations 2, 4, and 8 have significantly higher average workloads (per month), because they contain more tasks and more time-consuming tasks.

Finished products obstruct workspace at the production line (4), which means that workstations cannot start because the space is occupied. For example, workstations 7 and 8 are sometimes occupied with finished products. The cause for this issue is that the forklift operator does not pick up the finished products (5) since pick-ups happen randomly which do not match with the output of the production line (6).

The TU line experiences a shortage of materials (7), because stock levels are not correct (8). There are two reasons for this problem. The first reason is that production line workers do not accurately register used components (9) as they do not register damaged components, they do not routinely check their usage, and other production lines use components of the TU line but do not register it. The second reason is that suppliers deliver parts too late (10), as they experience their own issues.

A systematic shortage of production line operators at the production line (11). The number of production line operators is between 2 and 5 and they must utilize 8 workstations, which means that not all workstations are staffed during the process.

To conclude the discussion of the problem cluster, the main effect 'high average throughput time' is caused by several potential core problems and we analyze these causes. Problem 10 is difficult to control for the company, as the root causes originate with the suppliers, and agreements with the suppliers cannot be changed in the short term. Therefore, this problem is irrelevant. Problem 6 does not occur frequently, which makes this problem not relevant enough to research. Problem 9 is already researched within the company (and still ongoing), which makes this problem irrelevant. Fourth, problem 11 is difficult to solve in the short term as the hiring market is highly competitive and scarce where good mechanics are difficult to find. We choose problem 3 as the core problem of our research because it contributes to the solving the action problem and it is influenceable by the company. Thus, the core problem is identified as:

'The workload between workstations is not balanced'

The norm & reality is described by the problem owner (Trioliet engineer) as 'the average throughput time (per month) at the TU line should be reduced by 10%'.

1.4. Research objective & scope

This research aims to improve the throughput time of the TU line with 10% by solving the workload imbalance problem at the TU line.

To achieve this objective, we define the research scope to clearly outline the boundaries of the research. The research is conducted at the manufacturing facility in Oldenzaal. The manufacturing facility in Oldenzaal has multiple production lines for the assembly of multiple products, for example: silage cutters, mixed feed wagons and other feed systems. We look at the TU line (production line for silage cutters) and narrow the research towards re-allocating tasks across workstations to improve the workload balance at the TU line.

To conclude, the research scope is focused on changing the task distribution across workstations to improve the workload balance at the TU line, within the manufacturing facility in Oldenzaal.

1.5. Research limitations

The research is limited by the availability of data within the company, as throughput time of the TU line and workload of workstations were never collected.

1.6. Research questions

Having established the research motivation, core problem, research objective and research scope, we can formulate the main research question as: '**how can Trioliet reduce the average throughput time by solving the workload imbalance at the TU line?'**. We answer the main research question by answering 4 research questions and we discuss them in different chapters.

Chapter 2 Current situation at the TU line

Chapter 2 aims to formulate a better understanding of the current situation at the TU line. Sub-question 1.1 explains the product, production process (task distribution) and the current layout of the TU line. Sub-question 1.2 describes the current performance of the TU line by calculating the workload of the workstations and the process times of the product variants. Sub-question 1.3 identifies the bottleneck of the process.

1. What is the current situation at the TU line?
   - 1.1 What is the current production process of the TU line?
   - 1.2 What is the current performance of the TU line?
     - a. What is the current average workload at the workstations (per month)?
     - b. What are the current process times at the TU line?
   - 1.3 What is the current bottleneck at the TU line?

Chapter 3 Literature review into workload balancing
Chapter 3 seeks to understand the concept of workload balancing and assembly lines. Sub-question 2.1 describes the different types of assembly lines and Sub-question 2.2 describes the classifications of line balancing problems and their solution approach.

2. What is currently known about assembly lines and workload balancing in the literature?
   2.1 How can assembly lines be characterized?
   2.2 What is known about workload balancing at assembly lines?
      a. How can workload balancing problems be characterized?
      b. Which methods for solving workload balancing problems are discussed in the literature?

Chapter 4 Solution approach
Chapter 4 aims to formulate a solution approach for the balancing problem at the TU line, based on acquired knowledge from Chapters 2 and 3. Sub-question 3.1 describes the characteristics of the TU line, found in the literature, and Sub- question 3.2 describes a solution for the balancing problem at the TU line.

3. How can the theory be used to improve the workload balancing problem at the TU line?
   3.1 What are the characteristics of Trioliet's balancing problem?
   3.2 What is a suitable solution approach for solving the balancing problem of Trioliet?

Chapter 5 Results
Chapter 5 executes the proposed approach and shows the results of the research, where alternative solutions are compared with the current situation. Sub-question 4.1 chooses the best solution for Trioliet's problem with performance measures for 'workload balance'. Sub-question 4.2 compares the proposed solution with the current situation at the TU line, whereby the task and workload distribution are compared. Sub-question 4.3 calculates the throughput time improvement of the proposed solution.

4. How do alternative solutions perform compared to the current situation?
   4.1 What is the best solution?
   4.2 How does the proposed solution compare with the current situation?
   4.3 What is the throughput time improvement of the proposed solution?

Chapter 6 Conclusion and recommendations
We conclude the research findings and give recommendations to the company on improving the workload balance at the TU line.

2. Current situation at Trioliet

This chapter answers the question ‘what is the current situation at the TU line?’. Section 2.1 describes the product and the current production process of the TU line, including discussing the current task distribution. Section 2.2 describes the current performance of the TU line, whereby the current workload distribution is discussed. Last, Section 2.3 discusses the bottleneck at the TU line.

2.1. Production process

This section describes the production process of the TU line generally, where general tasks at workstations are explained. Detailed tasks are not explained but shown in Appendix A.1. Chapter 4 uses the detailed list of tasks to create a more detailed solution for Trioliet's problem.

Subsection 2.1.1 describes the product variants that the TU line assembles. Subsection 2.1.2 describes an overview of general tasks at the workstations at the TU line (current task distribution). Subsection 2.1.3 describes the production flows at the TU line and Subsection 2.1.4 gives a description of the layout of the TU line.

2.1.1. Products at the TU line

The TU line assembles the Turbobuster, which is a machine that cuts dry grass into squared blocks. Figure 4 shows a Turbobuster (abbreviation ‘TU’). The product is sold in different sizes that are classified into two groups: Standard TUs and TU180-XL. The standard TUs consists of 4 different frame heights (115, 145, 170 and 190 cm). The TU180-XL is 180 cm high and a bit wider than the standard TUs.

Figure 4: Turbobuster (Trioliet.com)

Additionally, these products have two variations: standard or configured. The standard variant of a product does consist out of a main frame, hose system and knife system. The main frame is the central component of the product. The knife system cuts silage and the hose system allows for the movement of the knife system (and knives). The configured variant of a product does also consist of a main frame, hose system and knife system, but it is also complemented with additional functions, called options, and/or with an electrical system. To summarize, there are 4 product variants of the TU.

1. Standard variant of standard TUs
2. Standard variant of TU180-XL
3. Configured variant of standard TUs
4. Configured variant of TU180-XL

2.1.2. Production process (current task distribution)

This subsection explains general tasks at the workstations to create a general understanding of the production process and the current task distribution at the TU line. Figure 5 shows the current task distribution of the TU line with general tasks.

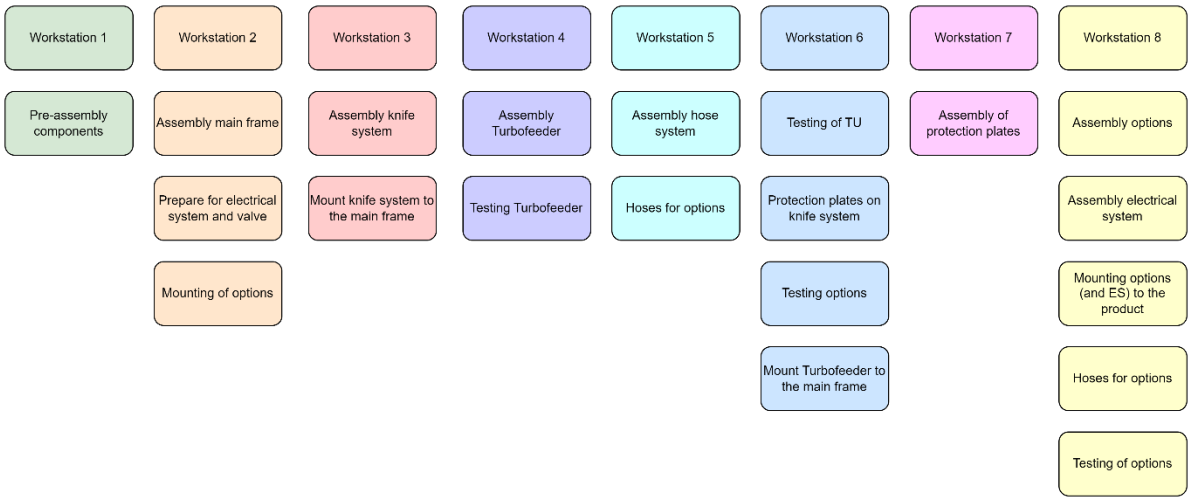


Figure 5: Current task distribution of the TU line with general tasks

Workstation 1
Workstation 1 pre-assembles two components for the main frame: walking charts and coupling plates. Coupling plates are used to move the product during the production process. Walking charts are used for attaching the knife system in workstation 3.

Workstation 2
Workstation 2 has 3 tasks: assembly of the main frame, preparation for the electrical system (and valve) and the assembly of options. The assembly of the main frame considers the assembly of multiple components to the main frame. For example, walking charts are mounted to the main frame in workstation 2.
The second task is 'preparing for electrical system (and valve)', where small components for the electrical system and valve are installed on the main frame. The third task 'mounting of options', consists of mounting the options hydraulic push-off unit (HPU) and mechanical push-off unit (MPU) to the product.

Workstation 3
Workstation 3 assembles the knife system and mounts this system to the main frame of the TU.

Workstation 4
Workstation 4 has two tasks, which are the assembly and testing of the option Turbofeeder (TF). The Turbofeeder is calibrated during the testing.

Workstation 5
Workstation 5 has two tasks: assembly of the hose system and attaching hoses for options. The assembly of the hose system includes installing hoses to the knife system and the main frame. Last, the hoses for options are attached to the product, which includes the hoses for the options HPU and/or TF.

Workstation 6
Workstation 6 has four tasks: testing of the TU, testing of options, assembly of the protection plates on the knife system and attaching the Turbofeeder to the main frame. Testing of the TU consists of checking the basic movement of the TU and knife system. Further, the options (HPU, MPU, TF) that were assembled in workstations 2 and 4 are tested at this workstation. Then, protection plates are assembled to the knife system. Last, the Turbofeeder is installed on the main frame.

Workstation 7
Workstation 7 assembles the protection plates on the product. When finishing workstation 7, a standard variant of the product (TUs/TU180-XL) leaves the production process and a configured variant of the product (TUs/TU180-XL) continues to workstation 8.

Workstation 8
Workstation 8 has 6 tasks: the assembly of options, assembly of the electrical system (ES), mounting the options to the product, mounting the ES to the product, attaching hoses of the options to the product and testing of the options (and ES).

The product can consist of many different options and they are assembled and mounted at this workstation. There are two types of options: pre-assembled options and unfinished options. Pre-assembled options only must be mounted on the product, and unfinished options must first be assembled and then mounted to the product. Further, the task 'hoses for options' is performed for hydraulic options, that work with the hose system (from workstation 5). Moreover, the electrical system is separately assembled and then mounted to the product.

Lastly, the options and the electrical system (ES) are tested. It conducts two types of tests, depending on the presence of an electrical system on the product. The first test only checks the working of separate options (if ES is not installed to the product). The second tests the working of the whole machine (if ES is installed to the product).

2.1.3. Production flows
This subsection discusses two parts: type of tasks at workstations and production flows of product variants.

Type of tasks at workstations
Workstations perform two kinds of tasks: basic and optional tasks. Basic tasks are always performed at workstations (regardless of production flow) and optional tasks are tasks that can be skipped or performed at a workstation.

Production flows
This part explains the production flows of product variants, specifically explaining when workstations are utilized for certain product variants and which tasks are performed at the workstations for certain product variants. Figure 6 shows the production flow of the product variants. Table 3 gives an overview of when the general tasks at workstations are executed for different production flows (product variants). The table has a legend, where performed tasks at a production flow are marked with an 'x' and optional tasks that can be skipped for a production flow are marked with an 'o'.

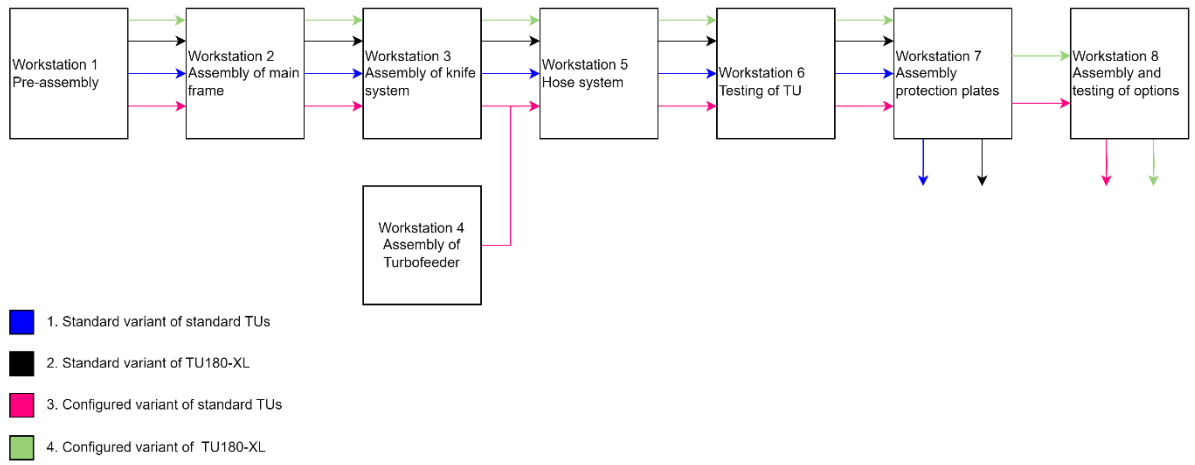


Figure 6: Overview of production flows of product variants

1. Standard variant of standard TUs
This flow goes through the workstations 1 to 7 (workstation 4 is skipped) and the product leaves the production line at workstation 7 (Blue line in Figure 6). As this flow goes through workstations 1 to 7, it performs only the basic tasks (Table 3).

2. Standard variant of TU180-XL
This flow goes through the workstations 1 to 7 (workstation 4 is skipped) and the product leaves the production line at workstation 7 (Black line in Figure 6). The standard TU180-XL considers one option, which is the HPU, as it is part of the standard variant of the TU180-XL, and this flow performs all the basic tasks throughout the process and performs four optional tasks for the HPU: assembly of the HPU in workstation 8, mounting of the HPU in workstation 2, attaching of the hoses for the HPU in workstation 5 and the testing of the HPU in workstation 6 (Table 3; tasks 4, 5, 12 and 15).

3. Configured variant of standard TUs
This flow goes through workstations 1 to 8 and workstation 4 can be skipped or not, depending on if the option Turbofeeder must be installed on the product (marked with 'o' in Table 4). The product leaves the production line at workstation 8 (pink line in Figure 6). Basic tasks at workstations 1 to 8 are always performed. Optional tasks related to the assembly, mounting and testing of options and preparation of the electrical system are performed at different workstations, depending on what options are assembled on the product. Multiple cases exist where workstations 2, 4, 5 and 8 perform or skip optional tasks (marked with 'o' in Table 3).

4. Configured variant of TU180-XL
This variant goes through workstations 1 to 8 (workstation 4 is skipped) and leaves the production line at workstation 8 (green line in Figure 6). Basic tasks at workstations 1 to 8 are always performed. Further, optional tasks at workstations 5 and 6 are always performed, since the HPU is considered a standard option and workstations 2 and 8 perform or skip optional tasks (marked with 'o' in Table 3).

Task Nr	Tasks description	Workstation	Flow 1	Flow 2	Flow 3	Flow 4
1	Pre-assembly components	1	x	x	x	x
2	Assembly main frame	2	x	x	x	x
3	Prepare for ES (and valve)	2			o	o
4	Assembly of options	2 and/or 8		x	x	x
5	Mount options to the product	2 and/or 8		x	x	x
6	Assembly knife system	3	x	x	x	x
7	Mount knife system to main frame	3	x	x	x	x
8	Assembly Turbofeeder	4			o	
9	Testing Turbofeeder	4			o	
10	Mount Turbofeeder to main frame	5			o	
11	Assembly hose system	5	x	x	x	x
12	Hoses for options	8 and/or 5		x	o	x
13	Testing of TU	6	x	x	x	x
14	Protection plates on knife system	6	x	x	x	x
15	Testing options	8 and/or 6		o	x	x
16	Assembly protection plates on the product	7	x	x	x	x
17	Assembly electrical system (ES)	8			o	o
18	Mounting ES to the product	8			o	o

Table 3: Production tasks for the different production flows

2.1.4. Layout of the TU line

Figure 7: Layout sketch of the TU line

The TU line has a 'single line' layout where workstations are positioned next to each other and operators work from all sides of the product (two-sided layout). Figure 7 shows a sketch of the current layout of the TU line. The layout of the TU line consists of 3 parts: inventory, worktables/frame holders and testing places. Inventory is positioned at or close to the necessary workstations. For example, option components are placed at or near workstation 8. Further, worktables or frame standards are positioned at workstations 2, 3, 4 and 8. Workstation 2 uses a rotational standard that can rotate the main frame of product. Workstation 3 assembles the knife system on a frame table. Workstation 4 assembles the Turbofeeder and tests the Turbofeeder in a cage (for safety). Workstation 8 assembles the options at a large table. Next, small tables are positioned at workstations 1 and 2 for the assembling of small components of the main frame. Last, the line has 2 testing places at workstations 6 and 8.

2.2. Performance of the TU line

This section answers the question 'what is the current performance of the TU line?'. Subsection 2.2.1 determines the current workload distribution at the TU line by calculating the workload of workstations. Subsection 2.2.2 calculates the average process time of the product variants.

2.2.1. Current workload distribution

In this research, we calculate the workload at workstations over a period of a month and define workload as 'the sum of the processing times of the tasks assigned to the workstation' (Dinler, 2021). The workload is determined by answering two questions:

1. What is the processing time of individual tasks?
   We use a specific list of tasks from observations and informal interviews. Further, we collect the processing time of individual tasks by conducting informal interviews with operators. The processing time of tasks can be different depending on the product variant that the TU line produces.

2. How often are tasks performed at a workstation in a month?
   We collected the frequency of tasks of different product variants (in a month) through sales numbers from 2022.

We calculate the workload of each workstation (per month) by adding up the process time of each task and how often each task occurs at a workstation. The calculation of the workload is done with the actual data from the company and Appendix A.1 shows the frequencies and task times of tasks, where we include a multiplication factor 'x'. Appendix E shows the multiplication factor 'x'.

Figure 8 shows the average workload at workstations (in hours per month) in the year 2022. We observe from Figure 8 that the calculated workload in general over a month is quite low, and this could be due to either the collected process times of tasks being on the lower side or the frequency of tasks in the year 2022 not being very high. For example, we see that workstation 2 (the highest workstation) has a workload of 81.25 hours per month and that is 20.31 hours per week, and 4.06 hours per day.

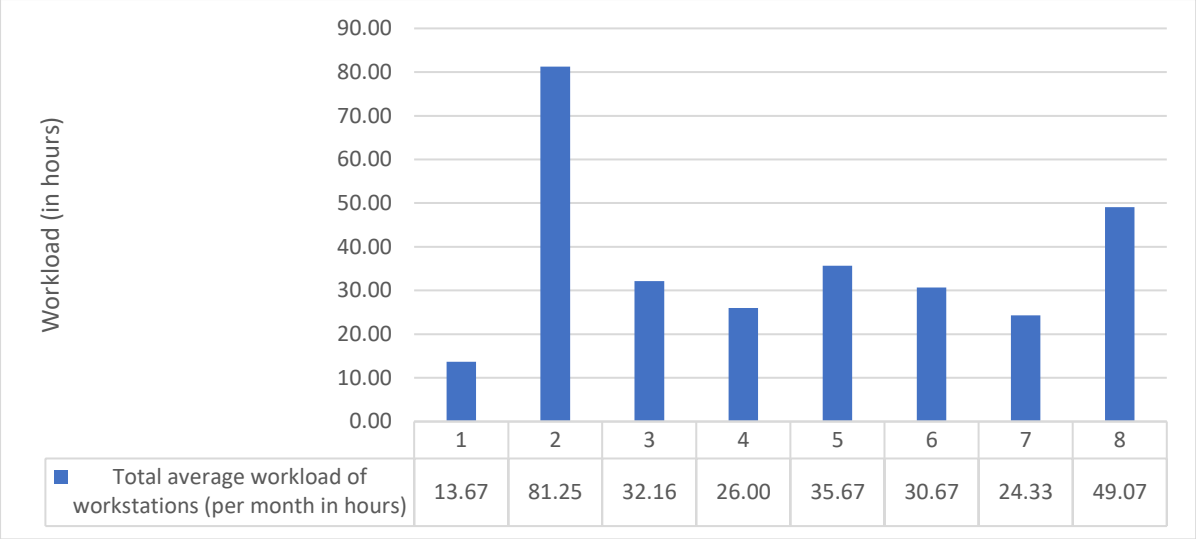


Figure 8: Current workload distribution at the TU line

2.2.2. Process time of products and total average process time per month

In this research, we were not able to calculate the throughput time of the TU line or their products due to time constraints with the research, therefore we give an estimation into the process times of the product variants and the total average process time per month. The process times are calculated based on the given process times of tasks and the product demand of the year 2022. The TU line produces 4 product variants, as mentioned in Section 2.1.1. We use a factor 'x' to hide the actual process times of the product. Appendix E shows the multiplication factor 'x'.

- Standard variant of the standard TUs
  The estimated process time of the standard TU is 9.88 hours (for one single unit) and consists of the summation of processing times of tasks 1, 2, 5, 6, 10, 12, 13 and 15 (Figure 9). This value considers a factor 'x' described in Appendix E.

- Standard variant of the TU180-XL
  The estimated process time of the standard TU180-XL (one single unit) is 13.42 hours (for one single unit) and consists of the summation of the processing times of tasks 1, 2, HPU assembly and mounting, 5, 6, 10, 11, 12, 13, 14 and 15 (Figure 9). This value considers a factor 'x' described in Appendix E.

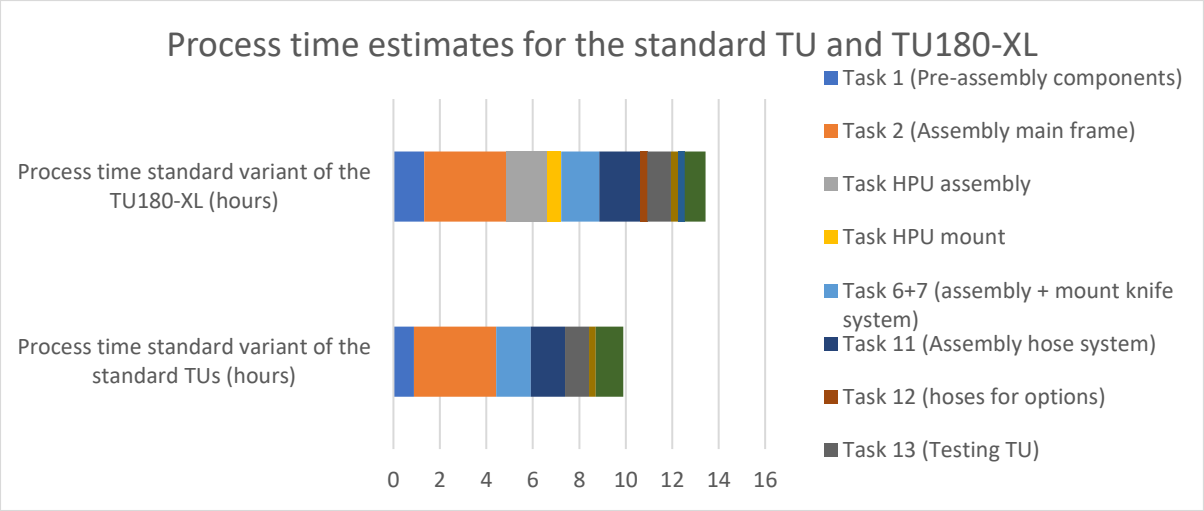


Figure 9: Process time estimates for the standard variant of the standard TUs and TU180-XL

- Configured variant of standard TUs

The process time of the configured variants of the standard TUs is more difficult to calculate since it is dependent on the number of options on a product and what options are attached to the product. The estimated process time of a configured TU is calculated with the process times of tasks for the configured TUs and the sales figures of configured TUs from the year 2022. Appendix A.2 shows the calculation of the processing time of configured TUs and TU180-XL in the year 2022.

Table 4 shows the average process time of the configured standard TUs for a different number of options and considers a factor 'x' for columns 'number of products produced in a year' and 'average process time of a single unit of a product with x options'. We conclude that the average process time of a configured variant of standard TU does not increase monotonically, because the products have different options that each have different process times. The average process time for a single unit of a configured standard TU is 19.04 hours (considering a factor 'x').

Configured standard TU with x number of options	Number of products produced in a year	Average process time of a single unit of a product with x options (in hours)
1	3.54	10.48
2	70.80	17.22
3	14.16	12.73
4	3.54	37.17
5	8.85	21.51
6	7.08	29.88
7 (With multiple electrical controls or with container transportation)	1.77	19.77
8 (With multiple electrical controls or with container transportation)	5.31	35.36

Table 4: Average process times of configured standard TUs with a different number of options

- Configured variant of TU180-XL

This estimated process time is calculated in the same way as the throughput time of the configured variant of the standard TUs. Process times of tasks for configured TU180-XLs and the sales figures of configured TU180-XLs from 2022 were used to calculate the process times.

Table 5 shows the average process time of the configured TU180-XLs for a different number of options and considers a factor 'x' for columns 'number of products produced in a year' and 'average process time of a single unit of a product with x options'. We can conclude that the process times do not increase monotonically, as products have different options with different process times. The average process time for a single unit of a configured TU180-XL is approximately 23.79 hours (considering a factor 'x').

Configured TU180-XL with x number of options	Number of products produced in a year	Average process time of a single unit of a product with x options (in hours)
1	23.01	16.82
2	14.16	22.57
3	3.54	24.78
4	17.70	23.58
5	24.78	25.47
6	17.70	28.25
7 (With multiple electrical controls or with container transportation)	3.54	34.87
8 (With multiple electrical controls or with container transportation)	1.77	34.34

Table 5: Average process times of configured TU180-XLs with a different number of options

To summarize, we have estimated the process times of the 4 product variants by using the sales figures from 2022 and the process times of tasks (from informal interviews with line operators). The average process time of a product at the TU line is calculated by using the average process time and demand of each product variant, and dividing it by the total demand: (9.88 * 550.47 + 13.42 * 3.54 + 19.04 * 115.05 + 23.79 * 106.2) / (550.47+3.54+115.05+106.2) = 23.29 hours. This value accounts for a factor 'x^2'. The total average process time per month is calculated by taking the entire workload of all products in a year and dividing by 12 months: (9.88 * 550.47 + 13.42 * 3.54 + 19.04 * 115.05 + 19.82 * 106.2) / 12 months = 815.13 hours per month. This factor accounts for a factor 'x^2'.

2.3. Bottleneck analysis

The current workload distribution of the TU line was calculated in Section 2.2 and this section determines the bottleneck of the TU line. A bottleneck is 'the most heavily loaded workstation', according to Pastor (2021).

Subsection 2.3.1 discusses the workload distribution of the TU line and Subsection 2.3.2 outlines the operator and inventory availability at the TU line. Further, Subsection 2.3.3 discusses the workstations with the highest workloads, which are workstation 2 and 8. Last, Subsection 2.3.4 provides a conclusion.

2.3.1. Workload distribution TU line

The current workload distribution at the TU line was calculated in Section 2.2.1 and three conclusions are drawn:

1. The current workload distribution is already quite in balance
2. Workstations 2 and 8 have a significantly higher workload (average per month in 2022)
3. Workstation 1 has a significantly lower workload (average per month in 2022)

2.3.2. Operator and inventory availability at the TU line

A workstation at the TU line can run when there is enough inventory available and when an operator is available.

- Inventory

Each workstation has its own inventory where small inventory is kept at the production line and is supplied from the warehouse. Large frames are supplied by the paint department. The availability of parts on the workstations depends on the supply of the warehouse and paint department.

- Operator availability

The TU line has 3 to 5 operators, and they are distributed across 8 workstations. This means that the TU line has a shortage of operators. As a result, the distribution of operators (operating hours) is constantly changing, because the head operator determines from time-to-time which workstations are left idle. A workstation without an operator is idle. In most cases, workstations 2, 3, 5, 6 and 8 are utilized, according to the head operator, which means that there are no other operators available for the other workstations.

2.3.3. Workstations

This subsection describes workstation 2 and 8, as they have the highest workloads at the TU line. The workload, operator and inventory availability, and influence on other workstations are explained.

Workstation 2

- Workload

The main reason for the high workload in workstation 2 is that many important subtasks are performed as preparation for mounting the knife system and hose system to the mainframe, which takes in total around 80 hours per month. These subtasks are combined into task 2 'assembly main frame'.

- Operator and inventory availability

Workstation 2 is operated by one operator and the main frame is supplied from the warehouse. If the main frame is not supplied or if there is no operator available, then the workstation is idle.

- Influence on other workstations

Workstation 2 is dependent on workstation 1 and influences workstation 3 to 8. We only discuss the dependencies of workstation 2 to workstation 1 and 3.

Workstation 2 depends on workstation 1. Workstation 1 pre-assembles two components for the main frame, independently, meaning that workstation 1 can continue uninterrupted as long as there is manpower and materials. workstation 1 is often left idle since its workload is much lower than that of workstation 2.

Workstation 3 is influenced by workstation 1 and 2. Workstation 3 is manned by 1 operator, and it produces the knife system that goes on the product. The assembly of the knife system is independent, but it still needs to be installed on the product and this action depends on workstation 1 and 2. The workload of workstation 2 is much higher than that of workstation 3, resulting in workstation 3 having to wait for workstation 2 to complete its tasks. Additionally, when workstation 1 does not complete its pre-assembly, the knife system cannot be mounted to the product as workstation 2 could not finish its tasks. When workstation 1 or 2 is not finished, operators decide to set aside the finished knife systems and proceed with the assembly of new knife systems. The finished knife systems are installed on the machine when the product is ready from workstation 2.

Workstation 8

- Workload

The main reason for the high workload is that there are many options assembled, mounted to the product and tested at this workstation.

- Operator and inventory availability

Workstation 8 is operated by only the head operator, as he is trained to assemble, mount and test options. This workstation has a wide range of inventory for the different options, like frames and other components. If the head operator or inventory are not available, the workstation is idle.

- Influence on other workstations

Workstation 8 is semi-independent as some options can be assembled independently but workstation 8 must wait until workstation 7 has finished. Then, workstation 8 can start mounting options to the product. The workload of workstation 7 is much lower than in workstation 8, causing more products to enter workstation 8 than can be processed, resulting in an accumulation of unfinished products in workstation 8.

2.3.4. Bottleneck analysis conclusion

The bottleneck analysis explained the TU line's workload distribution, available operating hours and inventory. Three conclusions can be drawn.

1. The current workload distribution is already quite in balance where workstations 2 and 8 have higher workloads, workstations 3 to 7 are somewhat in balance and workstation 1 has a lower workload.

2. TU line distributes 3-to-5-line operators across 8 workstations. Further, each workstation has its own inventory, and their supply depends on both the warehouse and paint department.

3. Workstations 2 and 8 were discussed in terms of workload, operator and inventory availability, and their influence on other workstations. The workload at workstations 2 and 8 is high because they perform many tasks. However, these workstations are only operated by a maximum of one line

operator. Thus, the capacity of these workstations is not well adjusted to their higher workloads. Therefore, we need to tactically reduce the workloads of these workstations by decreasing the number of tasks assigned to them.

2.4. Conclusion

Chapter 2 answers the question ‘what is the current situation at the TU line’.

Section 2.1 describes the TU line and their products. The TU line is an assembly line that produces the product 'silage cutter', called ‘Turbobuster’. There are four product variants of the Turbobuster: a standard variant of standard TUs, a standard variant of TU180-XL, a configured variant of standard TUs and a configured variant of TU180-XL. The TU line has 8 workstations and the current task distribution of the TU line shows which tasks are performed at the workstations. We see from the production flows that the workstations perform different tasks based on the product variant. Furthermore, we see that workstation 2 has a large task for preparation and workstation 8 contains many tasks for assembly, mounting, and testing of options. Lastly, the TU line has 2-to-5-line operators and the layout of the TU line is organized in series, considering working and testing places, and has its own inventory.

Section 2.2 determines the performance of the TU line, where the current workload of the workstations and process times of product variations were calculated. Furthermore, the total average process time (per month) at the TU line has been determined at 260.42 hours.

Section 2.3 conducted a bottleneck analysis, where the current workload distribution and workstations 2 and 8 were analyzed. The current workload distribution shows that the production line is already quite in balance, where workstations 2 and 8 have significantly higher workloads, workstations 3 to 7 are in balance and workstation 1 have a significantly lower workload.
Next, workstations 2 and 8 are explained where workload, operator and inventory availability, and relation with other workstations are outlined. Workstation 2 is operated by a maximum of one line operator and has a high workload since it performs many important subtasks as preparation for mounting the knife system and hose system to the mainframe. Workstation 2 depends on workstation 1, where two components for the main frame are assembled, and workstation 2 influences the following workstations 3 to 8. Workstation 8 is only performed by the head operator and has a high workload since it assembles, mounts and tests many options for the product. Workstation 8 is only influenced by workstation 7, where the main product comes from.

3. Literature

This chapter describes the insights from the literature review to answer the question 'what is currently known about assembly lines and workload balancing in the literature?'. Section 3.1 describes the characteristics of assembly lines. Section 3.2 explains different types of balancing problems and Section 3.3 discusses the solution methods for solving balancing problems. Further, Section 3.4 discusses the performance measures for workload balancing and Section 3.5 concludes this chapter.

3.1. Classification of assembly lines

Boysen (2006) defines an assembly line as 'a flow-oriented production system where the productive units perform operations, stations are aligned in a serial manner'. In the literature, similar characteristics are used for classifying assembly lines. Boysen (2007), Saif (2014) and Becker (2004) describe 7 characteristics to classify assembly lines: product, workflow, layout, objectives, tasks times, level of automation and line of business. Table 6 shows the classification of assembly lines.

Classification of assembly lines			
1. Product	Single model	Mixed model	Multi model
2. Workflow	Paced	Unpaced synchronous	Unpaced asynchronous
3. Layout	Series vs Parallel	One or Two-sided	U-shaped
4. Objectives	Minimize workstations	Minimize cycle time	Maximize efficiency
5. Tasks time	Deterministic	Variable	Stochastic
6. Level of automation	Manual	Automated	
7. Line of business	Automobile production	Other examples	

Table 6: Classification of assembly lines (Boysen, 2007)

- Product

The assembly line is determined by the products that are produced at the line and there are three types of assembly lines: single, mixed and multi lines. Single lines produce one kind of product. Mixed lines produce multiple variants of a product and multi lines produce different products on the assembly line. (Becker, 2004)

- Workflow

There are two types of workflows: paced and un-paced assembly lines. (Saif, 2014) Paced assembly lines deal with workstations that have similar deterministic cycle times. Un-paced assembly lines deal with workstations that have their own individual cycle times therefore workstations deal with idle times. There are two ways that un-paced assembly lines move their products: synchronously and asynchronously. In synchronous way, all workstations move the workpieces simultaneously. In asynchronous way, workstations move the workpieces individually.

- Layout

Layout means the physical positioning of workstations and it can be categorized into various kinds: serial lines, parallel lines, one-sided lines, two-sided lines and U-shaped lines. (Saif, 2014)

A serial line deals with workstations that are arranged consecutively, where the product is moved from one station to the next. Parallel assembly lines are designed to divide workloads among workstations. (Saif, 2014)

One-sided line is designed in a way that the product is worked on from one side. A two-sided line is designed so that the product is worked on from multiple sides and divides tasks within a station; therefore, fewer stations are required. (Saif, 2014)

A U-shaped layout is unique because the entrance and exit of the assembly line are at the same position. (Ağpak, 2011) This layout can reduce the number of workstations and reduce idle times with buffers. (Saif, 2014) Figure 10 shows the U-type line design.

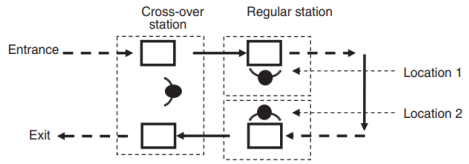


Figure 10: U-type line design (Ağpak, 2011)

- Objectives

An assembly line can have single or multiple objectives. The most common objectives are minimizing the number of workstations, minimizing cycle times and maximizing the production line efficiency.

- Task times

Assembly lines deal with task times that are deterministic, variable or stochastic. (Becker, 2004) Task times are deterministic if the expected variance of task times is sufficiently small, in cases of simple similar tasks and highly reliable workstations. However, task times are variable or stochastic. Variable task times deal with certain variations, for example variable work rates of operators or changes in the number of tasks performed at workstations. Stochastic task times deal with uncertain variations, for example unreliable workstations where a machine breaks down.

- Level of automation

Boysen (2007) discusses two types of lines: manual lines and automated lines. Manual lines are often subjected to stochastic task times due to the performance of operators. Automated lines are implemented for hostile environments or financially incentives, where a robot is economically cheaper and more precise.

- Line of business

The business sector influences how the assembly line operates. For example, automotive companies often deal with large product components and these lines have parallel workstations so that operators can work simultaneously on the same workpiece. (Boysen, 2007) Further, task times in automotive industries are often influenced by the work conditions of operators, which are often determined by the trade unions are employers.

3.2. Assembly line balancing problems

This section discusses assembly line balancing problems (ALBPs). Subsection 3.2.1 introduces assembly line balancing and Subsection 3.2.2 discusses the classification of ALBPs.

3.2.1. Introduction to assembly line balancing

Assembly line balancing (ALB) is defined as ‘arranging individual processing and assembly tasks at the workstations so that the total time required at each workstation is approximately the same’. (Kiran, 2019) The concept of assembly line balancing was introduced by Ford Motors in 1940 and this core problem is known as the simple assembly line balancing problem (SALBP). Later, more complex assembly line balancing problems emerged.

With the knowledge from Section 3.1 about line characteristics and from this subsection about the definition of assembly line balancing, we further explain the classification of the ALBPs in Section 3.2.2.

3.2.2. Classification of ALBPs

Assembly line balancing problems can be divided into two groups: simple assembly line balancing problem (SALBP) and general assembly line balancing problem (GALBP). (Becker, 2004) These ALBPs are further classified with their objectives and constraints. This subsection is subdivided into 4 parts: simple assembly line balancing problem (SALBP), general assembly line balancing problem (GALBP), objectives and constraints.

Simple assembly line balancing problem (SALBP)

A SALBP describes the simplest variant of a balancing problem because it makes simplifying assumptions. A disadvantage with these simplifying assumptions is that they do not always reflect the realistic situation of an assembly line. Boysen (2021) describes the nine assumptions of a SALBP.

1. The assembly line produces one homogenous product
2. All tasks are processed in a predetermined mode, which means that tasks are carried out in a predetermined way.
3. Workstations have a fixed cycle times and they move the product in a paced way
4. A single serial assembly line is considered
5. The processing sequence of tasks is subject to precedence restrictions
6. Tasks have deterministic task times
7. Tasks have no other assignment restrictions besides precedence constraints
8. Tasks cannot be split among two or more stations
9. All stations are equally equipped with respect to machines and workers

General assembly line balancing problem (GALBP)

GALBPs are more complex balancing problems because they have fewer simplifying assumptions compared to the SALBPs. GALBPs can consider multiple products, different layouts, variable/stochastic task times and other line characteristics from Section 3.1. There are many GALB problems, for example:

1. Mixed model assembly line problem (MMALBP) considers the production of multiple variants of a product on the assembly line.
2. Multi model assembly line balancing problem (MuMALBP) considers the production of different products on the assembly line.
3. U-line balancing problem (UALBP) considers an assembly line with a U-shaped layout.
4. Multi manned assembly line balancing problem (MALBP) considers multiple operators at a workstation.
5. Others

Objectives

SALBP and GALBP consider 4 types of objectives: minimizing cycle time (type 1), minimizing the number of workstations (type 2), maximizing efficiency (type E) and finding a feasible solution (type F).

Constraints

ALBPs must deal with constraints that must be satisfied. SALBP does only consider precedence constraints, which was already stated in assumption 7 (Section 3.2.2). GALBP considers 3 different constraints, which are precedence, zoning and resource constraints.

1. Precedence constraints refer to the task sequence, as tasks cannot always be carried out in an arbitrary sequence due to technological and organizational conditions (Becker, 2004). Precedence constraints are visualized with a precedence graph where tasks can have direct or indirect predecessors. Figure 11 shows an example precedence graph from Becker (2004). For example, Task 7 has one direct predecessor (task 2) and one indirect predecessor (task 1).

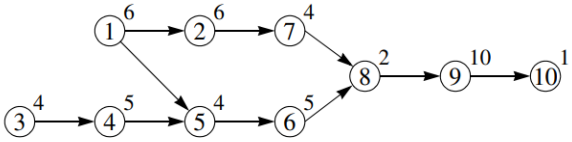


Figure 11: Precedence graph example (Becker, 2004)

2. Zoning constraints look at the compatibility and linkage of tasks. For example, tasks at a production line can be compatible or linked due to similar use of equipment or minimal workspace requirement. There are two types of zoning constraints: positive and negative zoning constraints. Positive zoning constraint assigns compatible tasks to the same workstation. Negative zoning constraint assigns tasks that are not compatible to different workstations. (Fathi, 2017)

3. Resource constraints are used when certain tasks require a specific resource, for example, a tool that is not available at all stations. (Alghazi, 2017)

3.3. Solution methods for ALBPs

Assembly line balancing problems are family of the combinatorial problems (problems that have a finite number of solutions) and are NP-hard, which means that the time needed to solve the problem grows exponentially with the size of the problem. Combinatorial problems are solved with exact or heuristic methods. Exact methods can provide a theoretical feasible optimal solution whereas heuristic methods provide a feasible solution with no optimality guarantee. (Geovanni, 2017) Subsection 3.3.1 discusses the exact methods and Subsection 3.3.2 discusses heuristic methods.

3.3.1. Exact methods

Exact methods are often impractical for solving middle-large to large balancing problems due to the complexity of the balancing problem. (Xu, 2023) We mention two methods that are used for balancing problems: integer programming and brand and bound method.

1. Integer programming

Fathi (2017) uses mixed integer programming (MIP model) for solving a balancing problem for straight and U-line configurations where the objective is to minimize the number of workstations. The model takes stochastic task times, cycle times, precedence and zoning constraints as input. A multi-attribute decision-making approach (TOTSIS) was chosen to compare different ALB problem scenarios in the research.

2. Branch and bound

Kellegöz (2012) uses a branch and bound algorithm to find an optimal balancing solution for a parallel multi manned assembly line. This method splits the problem into subproblems (branching) and eliminates subproblems based on priority values (bounding, also called pruning). For branching, this research uses the task-oriented branching scheme, where ‘sub trees’ are generated for different task assignments. Single eligible tasks are assigned to the earliest stations. For bounding, the ‘sub trees’ are prioritized, and a termination condition is used to eliminate ‘sub trees’.

3.3.2. Heuristics

A heuristic is a ‘problem specific approach that employs a practical method that often provides sufficient accuracy for the immediate goals’ (Lindfield, 2019) and it is used to solve large-scale balancing problems. There are two groups of heuristics in the ALB field: constructive and improvement heuristics. Constructive heuristic starts with an ‘empty’ solution and builds a solution, step by step, according to a set of rules. (Oliveira, 2009) Improvement heuristic starts with an initial (feasible) solution and improves it by applying successive small changes to the initial solution. (Oliveira, 2009)

Constructive heuristics

We are going to discuss 4 constructive heuristics, that use priority rules to assign tasks to workstations: Largest Candidate Rule (LCR), Killbridge and Wester (K&W), Rank Positional Weight (RPWM) and Number of Followers Heuristic (NFH).

1. Largest Candidate Rule (LCR)

The LCR heuristic aims to distribute workloads evenly among workstations. (Dung, 2019) Tasks are assigned from largest to smallest processing times, where they can only be assigned when the precedence constraints are satisfied and the sum of the processing tasks at a workstation do not exceed the (predetermined) cycle time of a workstation. This process is repetitive and exists when all tasks have been assigned to workstations.

2. Killbridge and Wester (K&W)

The Killbridge and Wester heuristic aims to minimize idle time. Tasks are assigned from lowest to highest precedence position, where tasks can only be assigned to workstations when the cycle time of the workstation is not exceeded, otherwise tasks are assigned to the next workstation. (Çelik, 2023) For example, task 10 with precedence 1 (column 1 in Figure 12) is assigned before task 14 with precedence 3 (column 3 in Figure 12). This process iterates until all tasks have been assigned to the workstations.

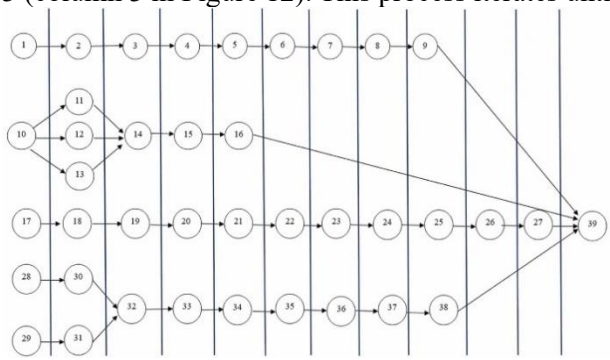


Figure 12: Column arrangement for K-W method (Çelik, 2023)

3. Rank Positional Weight Method (RPWM)

RPWM is a commonly used method for solving assembly line balancing problems, which combines the LCR and K&W heuristic. In RPWM, tasks are assigned to workstations based on rank positional weights (RPW) (Çelik, 2023). RPW for each task is determined by adding the task times of subsequent tasks in the precedence diagram. For example, task 9 is followed by tasks 11 and 12, which means that the RPW value of task 9 is 0.5 + 0.12 = 0.62 (Figure 13).

Tasks are assigned to workstations based on RPW value, where the highest value is assigned first. This process is iterative and stops when all tasks have been assigned to workstations. If the sum of task times exceeds the cycle time of a workstation, then tasks are assigned to the next workstation in the process (Çelik, 2023).

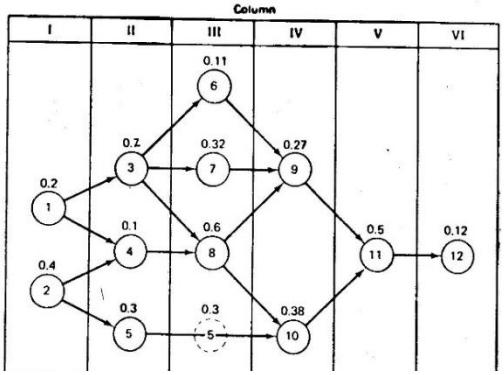


Figure 13: Precedence diagram RPWM method (Cuik, 2013)

4. Number of Followers Heuristic (NFH)

This heuristic assigns tasks based on the number of subsequent tasks, where the task with the highest number is assigned first. (Breginski, 2013) Tasks can only be assigned to a workstation when precedence relations are satisfied, and the cycle time of workstations are not exceeded.

Local search (improvement) heuristic

A local search heuristic, or improvement heuristic, is a method that iteratively improves a single solution, called current solution, by making small changes to it, resulting in a neighbor solution. (Glover, 2015). A set of neighboring solutions is called neighborhood. A neighbor solution is accepted with a search strategy. In this part, we explain the different local search strategies: Simulated Annealing (SA), Tabu Search (TS), Iterative Local Search heuristic (ILS), Hill Climbing (HC) and Steepest Descent (SD).

1. Simulated Annealing (SA)

SA is a stochastic approach for solving combinatorial problems and this method is used for large sized cases of ALBP. (Roshani, 2015) The method starts with an initial solution (current solution) and improves them iteratively by exploring neighboring solutions. SA evaluates a few movements (neighboring solutions) in the iteration, and it can accept or reject a neighbor solution. The neighboring solution is accepted if it doesn't decrease the score or if it passes a random check (when it does decrease the score). Thus, at the beginning SA does not always pick the move with the highest score and it gives non improving moves also a chance to be picked. In the end, it gradually only accepts improving moves. The result of this search strategy is that the end solution reaches a global maximum, which means that an optimum solution is reached.

2. Tabu Search (TS)

TS is a generalized local search procedure that uses 'memory' (information of historical search solutions) to prevent that solutions are visited multiple times. (Lapierre, 2006) TS starts with an initial solution and looks through all neighboring solutions. TS chooses the best neighboring solution, which is the move that leads to the highest value of the objective function. TS maintains a tabu list to avoid getting stuck in local optima. This list holds recently used objects, like a previous solution, value, move or entity, that are *taboo* to use for now. Moves that involve an object in the tabu list are not accepted. If you pick a too small tabu size, the TS can still get stuck in a local optimum. However, except solution tabu, if you pick a too large tabu size, you can get stuck by bouncing the walls. When a solution is accepted, it becomes the current solution, and this process is iterative.

3. Iterative local search heuristic (ILS)

Antoine (2016) presents an ILS for an assembly line rebalancing problem. The ILS meta-heuristic consists of four steps: initial solution generation, local search procedure, perturbation, and stopping criterion. The initial solution is generated randomly, and the local search procedure uses the swap or move operators to assign tasks to workstations randomly. A neighbor solution is accepted when the cycle time value is improved compared to the current solution. A perturbation (disturbance) mechanism is used to avoid a local optimum, where two tasks are randomly swapped. The ILS heuristic stops if a fixed number of iterations is reached, or the cycle time value is smaller than the takt time of tasks.

4. Hill Climbing (HC)

Yuan (2012) discusses a two-sided assembly line balancing problem and he proposes a Late Acceptance Hill Climbing heuristic (LAHC). HC starts with an initial solution and tries to improve this solution by iteratively exploring neighbor solutions with operators. The neighbor solution is only accepted when it improves on the current solution. The iterative process ends when it reaches a local optimum or it is set on a specific number of iterations is reached.

5. Steepest Descent (SD)

Scholl (1996) describes the SD heuristic for simple assembly line balancing problems. SD is an approach to overcome a local minimum, so it ends in a global minimum. Therefore, it can accept neighbor solutions that do not improve the objective value (it can also worsen the objective value).

Split, move and swap tasks in line balancing problems

We discussed how tasks are assigned in constructive heuristics and the search strategies of improvement heuristics, in previous parts of this subsection. In this part, we discuss splitting of tasks, moving and swapping of tasks.

1. Splitting of a task into subtasks.

Grzechca (2015) looks at splitting tasks into subtasks, before assigning tasks to workstations, which increases the number of tasks to be assigned to workstations. These tasks/subtasks are assigned (inserted) to workstations, according to a priority rule (constructive heuristics).

2. Move a task and/or swap a task (operators for local search heuristics)

We discuss four research examples that used move or/and swap operators for solving balancing problems with local search heuristics.

Wu (2023) uses the local search operator 'move', where he moves a randomly selected task. The priority relations of the random task are assessed, where immediately preceding and succeeding tasks are found. The randomly selected task can be moved within this range of moves.

Lappierre (2004) uses Tabu Search that explores two neighborhoods. The first neighborhood explores solutions where tasks move from overloaded to near empty workstations. The second neighborhood explores solutions where tasks exchange (swap) between the selected workstation and other workstations.

Roshani (2015) has a Multi-manned Assembly Line Balancing Problem (MALBP) and uses a Simulated Annealing heuristic where tasks from a priority list are randomly assigned to workstations. The tasks consist of priority values and these priority values are swapped or moved.

Li (2019) has U-shaped disassembly line balancing problem (SUDLBP) and developed a MIP model, where he proposed an Iterative Local Search procedure (ILS). The procedure has two operators. First, a random task is moved (a task is deleted and inserted) to another workstation. Second, two random tasks are randomly swapped.

3.4. Performance measures for workload balance

In Sections 3.2 and 3.3, we have introduced ALBPs and described solution methods for ALBPs. This section describes the performance measures for 'workload balance'. The literature speaks about the concept of 'measure of balance' and we mention 4 objectives for measuring 'workload balance' from the sources Vanheusden (2020 & 2022), Huang (2006) and Kumar (2001).

The first objective 'min [maximum workload]', from Kumar (2001), aims to minimize the workload of the workstation with the highest workload. It is a popular objective as it indirectly aims to minimize the throughput time of a production line. Alternatively, the objective 'max [minimum workload]' aims to increase the workload of the workstation with the minimum workload, which does increase this workstation's utilization.

The second objective 'range' looks at the differences between the maximum and minimum workload of the workstations (Vanheusden, 2020). If the range value is zero, the production line is perfectly balanced. The goal is to minimize the range value, where the maximum workload is minimized and/or the minimum workload is maximized. Kumar (2001) mentions the same objective and refers to it as 'min [difference between maximum and minimum]'.

The third objective 'pairwise differences' looks at the differences between entities. In the context of measuring workload balance at production lines, the entities are for example: difference in number of tasks, duration of tasks or workloads of tasks. Kumar (2001) and Huang (2006) use the objective 'min [average pairwise difference]', where it looks at the average workload difference between workstations (APW_D). Equation 1 shows the APW_D function and the notation is as follows: NMP is the number of machines pairs. SMP is set of all machine pairs. u_n is the utilization of machine n.

$$\mathrm{AP}W_D = \frac{1}{NMP} \sum_{(m,n)\in SMP} |u_m - u_n| \qquad (1)$$

The fourth objective is the Mean Absolute Deviation (MAD) and it measures workload balance by using the workload of machine m and the mean workload across all workstations. Nguyen (2013) states that the mean absolute deviation (MAD) is a good metric to measure balancing objectives, but Vanheusden (2020) states that MAD is not always able to minimize peaks in workload. Vanheusden (2020) investigates how the workload should be distributed across time slots during the day. Figure 14 shows two examples with the same objective value (MAD), where example 'b' was able to minimize the peak and example 'a' was not able to minimize the peak. The MAD value is the same for both examples because the 'extra workload' of one time slot is cancelled out by the 'less workload' of the other time slot. Equation 2 shows the MAD function and the following notation is as follows: M is the number of workstations, W_m is the workload of workstation m and $\bar{W}$ is the average workload across the workstations.

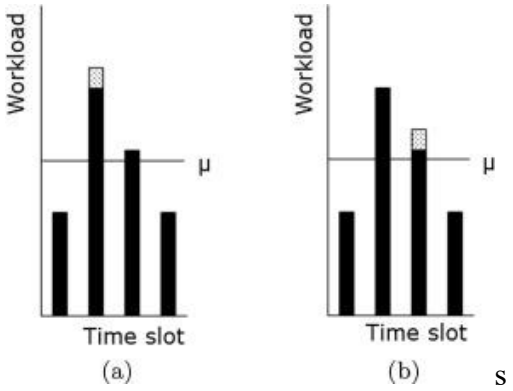


Figure 14: Example of different solutions with equal MAD. (Vanheusden 2020)

$$\mathrm{MAD} = \frac{1}{M} \sum_{m}^{M} |W_m - \bar{W}| \qquad (2)$$

3.5. Conclusion

The goal of this chapter was to answer the question 'what is currently known about assembly lines and workload balancing in the literature?'.

Section 3.1 determined the characteristics of an assembly line. An assembly line is characterized by 7 characteristics: the product, workflow, layout, objectives, tasks times, level of automation and line of business.

Section 3.2 discussed different types of assembly line balancing problems (ALBPs). There are two types of ALBPs: simple assembly line balancing problem (SALBP) and general assembly line balancing problem (GALP). The SALP considers many simplifications and does not always reflect the realistic assembly line. The GALP uses less simplifications and thus reflects reality better. Further, these problems

consider 4 types of objectives: minimize cycle time, minimize the number of workstations, maximize efficiency or finding a feasible solution. Next, ALBPs consider precedence constraints, which determine the sequence of tasks performed, and zoning constraints, which determine the arrangement of workstations.

Section 3.3 discussed the solution methods used for solving balancing problems. ALBPs are a family of combinatorial problems, and they are often solved with exact or heuristic methods. Exact methods are often not suitable for complex ALBPs as they require a long computational time. Heuristic is a practical approach, and it is often used for solving ALBPs. Two types of heuristics were discussed: constructive and improvement heuristic. Constructive heuristics solve the ALBP by starting with an empty solution and tasks are assigned based on priority rules. Further, improvement heuristics start with an initial solution and try to find a better (neighboring) solution, where tasks are either moved or swapped.

Section 3.4 discusses five objective measures for workload balance at production lines. The first objective measures the workstation of the highest or lowest workload. The second objective measures the difference between the workstations with the highest and lowest workload. The third objective measures the average pairwise differences between workstations. The fourth objective is the mean absolute deviation, which measures the absolute deviation from the mean workload across workstations.

4. Solution approach

This chapter answers the research question: ‘what is a suitable solution approach for solving the balancing problem of Trioliet?’. This chapter builds upon the findings of Chapters 2 and 3. Chapter 2 analyzed the current situation at the TU line and Chapter 3 describes the findings from the literature research. This chapter formulates a solution approach for Trioliet’s problem. Section 4.1 describes the characteristics of Trioliet’s balancing problem and Section 4.2 formulates the solution approach generally. Sections 4.3 and 4.4 discusses the specifics of the solution approach. Then, Section 4.5 describes suitable objective measures for Trioliet’s problem. Next, Section 4.6 describes the zoning and precedence constraints of the research. Finally, Section 4.7 concludes the chapter.

4.1. Balancing problem at the TU line

Section 3.2 discussed the assumptions of a simple assembly line balancing problem (SALBP). This section compares the balancing problem at Trioliet with the assumptions of SALBPs from the literature. The TU line has the following characteristics:

1. The line produces 4 product variants
2. All tasks are processed in a predetermined way (*)
3. Workstations do not have a fixed cycle time and they move the product in an un-paced asynchronous matter
4. The line has a serial layout where the product is assembled from multiple sides (*)
5. The processing sequence of tasks is subject to precedence restrictions (*)
6. Tasks have variable processing times
7. Tasks have two assignment restrictions: precedence and zoning constraints (*)
8. Tasks can be split among two or more workstations
9. Workstations are not equally equipped with respect to resources or workers.

The execution of tasks in a predetermined mode (2), a serial layout (4), two assignment restrictions (7) and processing sequence of tasks subject to precedence restrictions (5) are aligned with the assumptions of a SALBP. These characteristics are marked with an asterisk '*'.
The other characteristics (1, 3, 6, 8, 9) of the TU line do not align with the assumptions of a SALBP. Therefore, we classify Trioliet’s balancing problem as a general assembly line balancing problem (GALBP).

4.2. Solution approach

In the literature, we have seen that a line balancing problem can involve setting up a task allocation for a production line from scratch or improving an already existing line. The first balancing problem (the production line is redesigned from scratch) starts with an empty solution and these problems are solved with exact or heuristic methods. Our research does not consider exact methods due to time constraints of the research. Heuristic methods, more specifically, constructive heuristics methods are used to find a feasible solution, where tasks are assigned to workstations according to a set of priority rules. The second balancing problem ‘rebalancing’ is often solved with local search heuristics where alternative solutions are explored.

Trioliet's problem is more of a 'rebalancing problem' since they already have a current task allocation. However, this research is going to create task allocations (of the TU line) from scratch with a set of priority rules and then improve them with a local search heuristic. The reason for this approach is that we can generate multiple solutions for Trioliet's balancing problem.

The problem-solving approach for Trioliet's balancing problem consists of 2 steps: initial solutions and improving the initial solutions.

1. Initial solutions.
   The first step of the solution approach is to generate initial task allocation solutions, where we have the current task allocation of the TU line and where we design three task allocations for the TU line from scratch with a set of priority rules. Section 4.3 discusses the solution method, constraints and the iteration procedure of this solution approach step.

2. Improving solutions
   The second step of our solution approach is improving the initial task allocations with a local search procedure. Section 4.4 discusses the solution method, constraints and the iteration procedure of this solution approach step.

4.3. Initial solutions

The first step of the solution approach is going to initialize starting task allocations (initial solutions). Subsection 4.3.1 describes the solution method used in this research. Then, Subsection 4.4.2 discusses the constraints used in the generation of the initial task allocations. Last, Subsection 4.4.3 discusses the iteration procedure.

4.3.1. Solution method

Chapter 3 discussed constructive heuristics that generated initial task allocations from an empty solution. These heuristics select tasks to workstations, according to a set of priority rules. Different priority rules are used to select tasks based on ‘precedence relations’, ‘takt time’, ‘remaining assignable workload to assign’ or randomly

Our method selects tasks according to precedence position in the precedence diagram, where we release multiple tasks in each iteration. In Trioliet's case, we release at most 3 tasks per iteration (list of available tasks holds at most 3 tasks), because the tasks in the precedence diagram have at most 3 neighboring tasks. Then, we choose one task ‘randomly’ from the list of available tasks, because it is a simple way to select different tasks every iteration, resulting in different initial solutions. We choose not to select tasks based on other priority rules 'takt time', 'remaining workload to assign' or (completely) ‘randomly’, because of three reasons:

1. They can lead to infeasible solutions (precedence and maximum workload constraint cannot be met)
2. They can lead to overloading of the last workstations (precedence must be met)
3. They lead to the same tasks are selected in each iteration for every solution (we want to avoid this because we want to create different initial solutions).

Further, the selected task can be assigned to a workstation in multiple ways: randomly, ‘to the first possible workstation’ or ‘to the workstation with the highest to lowest available workload’. We choose to assign tasks to the first possible workstation, according to the precedence and maximum workload constraint, because tasks are less likely piling up at the last workstations or lead to infeasible solutions. Additionally, it is easier to move tasks, during the local search heuristic procedure, to the back of the line than the other way around due to the precedence constraints.

4.3.2. Constraints

Generating task allocations for the TU line from scratch involves three types of constraints: zoning, precedence, and maximum workload constraint.

- Zoning constraints

We set up zoning constraints of tasks before starting the iterative process of initializing the starting situations. With the zoning constraints, we combine tasks or subtasks into one task, so that they are performed at the same workstation. Section 4.6 explains the zoning constraints that are used for initializing task allocation solutions.

- Precedence constraints

Chapter 2 stated that the TU line assembles multiple product variants of the Turbobuster, which means that each product variant does have their own precedence relations. To simplify this problem, we set up a general precedence relation diagram that satisfies all product variants. Section 4.6 explains the precedence diagram that is used for generating initial task allocations.

- Choice of maximum workload of a workstation (per month)

We set up a maximum allowable workload for a workstation as we want to distribute the tasks across 8 workstations (and not assign all tasks to one workstation). In Chapter 2, we calculated the workload of workstations and the average workload of across workstations, which is 36.6 hours per month.

Setting up the maximum workload constraint for a workstation is a balancing act between the number of workstations and piling up tasks at the last workstations at the production line. When setting up a low maximum workload capacity at a workstation, it results in the last workstations getting overloaded with tasks. Otherwise, setting up a high maximum, results in less workstations are needed.

We set the maximum allowable workload for a workstation (per month) to 36.6 hours per workstation (per month) since it fills up all eight workstations and it is our aim workload for a workstation. Our solution has one exception, which is that workstation 2 always exceeds the maximum workload, because of the task 'main frame', which has a workload of 80 hours.

4.3.3. Iteration procedure

We start with an empty solution where no tasks are assigned yet and the procedure has 3 steps for each iteration: select task, assign task and stopping criterion.

1. Create a list of available tasks & select a task

We make a list of tasks based on precedence position, where at each iteration 'new' tasks become available (at most 3 tasks become available since tasks in the precedence graph have at most 3 neighbor tasks). From this list of available tasks, we select one task randomly.

2. Assing the chosen task to the first available workstation

The selected task is assigned to the first available workstation where the precedence constraints are satisfied, and the maximum workload of a workstation is not exceeded. The maximum workload of a workstation is set to 36.6 hours (per month) and the total required workload to allocate over the workstations is 292.81 hours per month (sum of all workloads of all tasks at the TU line).

3. Stopping criterion

The iteration of the iteration procedure stops when all tasks are assigned to a workstation.

4.4. Improving solutions

The second step of the solution approach is to improve the initial task allocations by exploring neighbor solutions. Subsection 4.4.1 describes the solution method and Subsection 4.4.2 explains the iteration procedure.

4.4.1. Solution method

In the second step of the solution approach, we want to bring changes to the initial task allocation solutions and Chapter 3 discussed different methods to solve these kinds of balancing problems. It discussed local search heuristics, which is a practical approach that cannot guarantee an optimal solution for solving rebalancing problems. With local search, we look at different search strategies (including being greedy or looking through all combinations), look at different operators, neighborhoods and accepting neighbor solutions. This subsection consists of 3 parts: operators & neighborhoods, search strategy and accepting neighbor solutions.

Operators & neighborhoods

Chapter 3 discussed that tasks could be moved or swapped for solving assembly line balancing problems with local search heuristics. This research uses both operators move and swap to improve the TU line's workload balance, thus it works with two neighborhoods.

- The first neighborhood (nh1) is for the 'move' operator, where we are going to look at moving a task from a selected workstation (with the highest workload) to another workstation with a lower workload.

- The second neighborhood (nh2) is for the 'swap' operator, where we are going to look at exchanging a task from a selected workstation (with the highest workload) with another workstation with a lower workload that has swap tasks that are smaller than the selected task from the selected workstation.

Search strategy

Chapter 3 introduced several local search heuristics, including Simulated Annealing, Tabu Search, Hill Climbing (HC) and Steepest Descent (SD). We are not going to use Simulated Annealing or Tabu Search in our research due to the time constraints of our research. HC and SD are similar methods, but SD explores all combinations to identify the best neighbor solution, whereas HC explores neighbor solutions in a "greedy" manner, where it immediately accepts the first improvement.

This research uses a hill climbing heuristic in a semi 'greedy' manner, where we work with two neighborhoods: move (nh1) and swap (nh2). We look for the first improvement within nh1 and nh2. Then, we accept the best solution out of these two first improvements of these neighborhoods.

Accepting neighbor solutions.

Neighbor solutions are accepted with the Mean Absolute Deviation objective (MAD), because of three reasons.

1. It is a simple interpretable function
2. It considers the workload of all workstations
3. It can minimize workload peaks, if all workstations do not lie above or below the mean workload (across workstations).

4.4.2. Iteration procedure

The iteration procedure has 4 steps: selecting a task, assigning a task, accepting a solution and stopping the iteration.

1. Select a workstation

We select the first possible workstation from high to low that can still be changed (selected workstation). Then we choose a task on the selected workstation from high to low that can still be moved or swapped (selected task)

2. Assigning tasks with operators in a neighborhood

As we have two operators, we work with two neighborhoods: nh1 (move) and nh2 (swap)

- Move
  The selected task from the selected workstation is moved to a workstation with a lower workload in order from lowest to higher (where precedence does allow it). The first possible improvement made with the move operator is chosen for the nh1 neighborhood.

- Swap
  The selected task from the selected workstation is swapped with another task from a lower workstation. The 'swap' workstation is chosen from lower to higher workload. Then, a swap task is chosen in order from lower to higher and this task should be smaller than the selected

task. The first possible improvement made with the swap operator is chosen for the nh2 neighborhood.

3. Accept solution

We accept the neighbor solution with the lowest objective value (MAD) out of both neighborhoods: nh1 and nh2.

4. Stopping criterion

This iteration procedure ends up in a local minimum and the procedure stops in two ways:

1. When the objective value cannot be improved anymore
2. When no swaps/moves can be performed with the tasks of the selected workstation: we only look at the highest and second highest workstation (in one iteration), otherwise the iteration is stopped.

4.5. Objective functions for 'workload balance'

Trioliet's problem aims to improve the workload balance at the TU line and Section 3.4 discussed five objectives to measure workload balance. This section goes into the advantages and disadvantages of using these objectives.

1. Minimize the maximum workload
2. Maximize the minimum workload
3. Minimize the difference between maximum and minimum workload
4. Minimize the average (pairwise) workload differences between workstations
5. Minimize the mean absolute deviation (MAD)

Objective (1) resolves the bottleneck workstation and objective (2) increases the workload of the idlest workstation. Objective (3) minimizes the gap between the bottleneck workstation and the idlest workstation. A downside of objectives (1), (2) and (3) is that they do not provide context about the other workstations outside of the extreme values (workstations). This means that the objective value could imply a good workload balance, but the distribution of the non-extreme workstations could be out of balance. Objective (4) cannot be used for Trioliet's problem because the utilization of workstations of the balancing solutions are not known at the TU line. Objective (5) looks at the differences of workload at workstations between the overall mean. This measure is easy to interpret and gives a clear workload distribution over the workstations. A downside of this measure could be that it is not always able to minimize peaks in workload, which means that different solutions could result in the same MAD value.

For Trioliet's problem, we use objectives (1), (2), (3) and (5) to measure the workload balance performance of the balancing solutions. Subsection 5.2.2 shows the results of the balancing solutions with these performance measures.

4.6. Description of constraints

Subsection 4.6.1 defines the specific list of tasks with zoning constraints that we need for our solution generation. Subsection 4.6.2 explains the precedence relations of the tasks at the TU line.

4.6.1. Zoning constraints

Chapter 3 stated that zoning constraints look at compatibility and linkages of tasks (Fathi, 2017). For example, a set of tasks require the same resources. This research uses zoning constraints to ensure that a set of tasks is performed at the same workstation. Chapter 2 explained general tasks at the TU line and we have established a detailed list of tasks in Appendix A.1, however this list does not take zoning constraints into account; therefore, we must make a new list of tasks that incorporate them.

The new list that includes zoning constraints is created by combining various tasks or subtasks so that they form one task. As a result, these individual (sub)tasks cannot be moved or swapped individually in

our solution method, but only ‘the combination’ can be moved or swapped. Table 7 shows the detailed list of tasks with zoning constraints, that are used for the solution generation.

Nr	Task description
1	Pre-assembly walking charts
2	Pre-assembly coupling plates
3	Assembly main frame
4	Prepare for ES (and valve)
5	Assembly knife system and mount knife system to main frame
6	Assembly Turbofeeder and testing Turbofeeder
7	Mount Turbofeeder to main frame
8	Pre-assembly control block
9	Mounting control block to TU/TU180-XL
10	Hoses
11	Hoses for options
12	Testing TU
13	Protection plates on knife system
14	Testing options
15	Protection plates on the product (and cleaning)
16	Assembly electrical system (ES) and electrical controls for options
17	Mounting ES to the product
18	Highlift cat3 assembly and mount
19	Highlift cat2 assembly and mount
20	Valve assembly
21	HPU separately sold
22	MPU separately sold
23	Highlift separately sold TU180-xl
24	Hydraulic push-off unit assembly TU180
25	Coupling systems
26	Valve mount
27	Sideshift
28	Toplink
29	Extension cord
30	MPU mount
31	HPU mount

Table 7: Detailed list of tasks including zoning constraints

4.6.2. Establishing precedence constraints

Chapter 2 stated that the TU line produces multiple variants of a Turbobuster. These product variants have their own precedence relations, because of the different components that are assembled and mounted to the product, which makes our balancing problem too complex. Thus, we simplify the precedence relations by setting up general precedence relations that account for all the product variants. We

use the specific list of tasks with zoning constraints (Table 7), which was established in Subsection 4.6.1.

Setting up general precedence relations results in that some tasks come later in the precedence position; despite that they could be performed earlier if they were related to a specific product variant. This holds for the mounting of options to the product (tasks 18, 19, 25-31) as there is a specific task sequence to mount options on the product.

We explain one example where we 'delayed' tasks in the precedence diagram due to generalization, which is the mounting of the 'sideshift'. We have two cases: a product with only the option 'sideshift' and a product with the options 'sideshift and highlift'. For the product with only the option 'sideshift', we can perform task 27 (mounting of sideshift) directly when task 3 has finished, thus the task sequence is 3 –> 27. However, when we have a product with a highlift and sideshift, we can only perform task 27 after task 18 as the highlift should be assembled and mounted to the product. As task 18 can only be performed after task 15, the task sequences become 15 –> 18 –> 27.

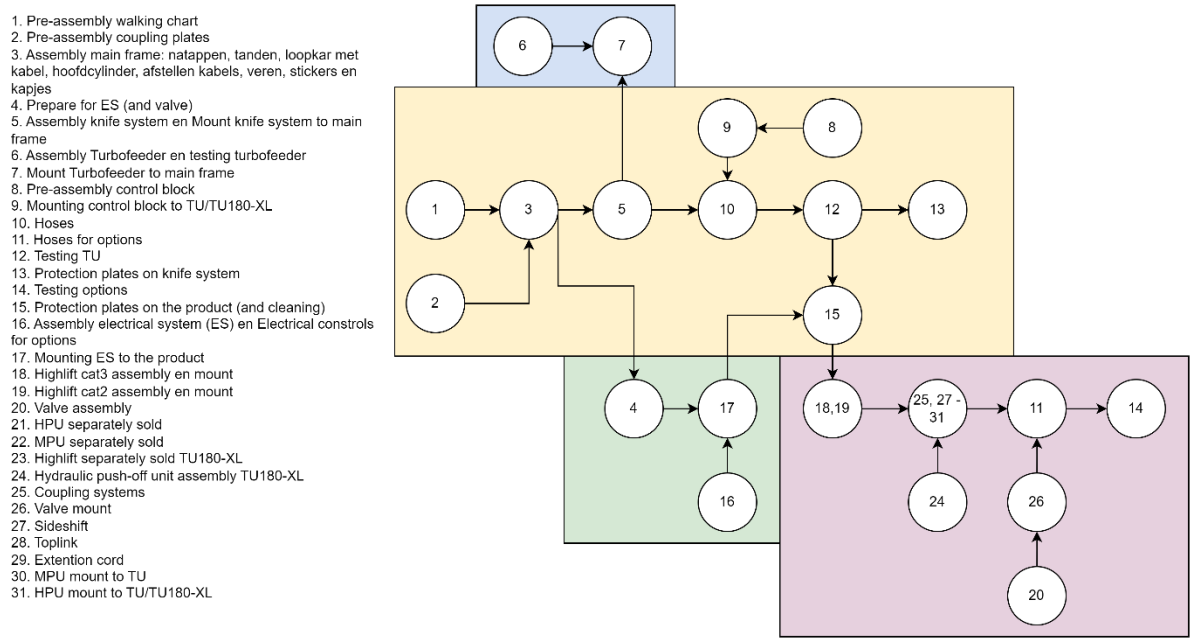


Figure 15: General precedence constraints

The precedence diagram (Figure 15) can be divided into four parts and these are shown with color blocks (yellow, green, blue and purple).

- Yellow block

The yellow block represents the tasks for a product without options. The process starts with tasks 1 and 2 (pre-assembly) and the main frame, knife system and hose system are assembled, mounted to the product and tested throughout the progression. The process finishes with tasks 13 and 15 (assembly of protection plates).

- Green block

When the product consists of an electrical system, tasks 4, 16 and 17 are performed. The ES must be mounted to the system before the assembly of the protection plates (task 15).

- Blue block

When the product consists of the option 'Turbofeeder', tasks 6 and 7 are performed. The Turbofeeder can only be attached to the product when the knife system is attached to the product (task 5 is finished).

- Purple block

The purple block represents the tasks for the assembly, mounting and testing of options. Options can only be mounted to the product when the assembly of the protection plates has finished (task 15). Tasks 21 to 23 are not included in the precedence diagram, as they are completely independent from other tasks. Further, tasks 20 and 26 are performed before task 11.

4.7. Conclusion

Section 4.1 compared the line characteristics of the TU line with the line characteristics of a SALBP. Trioliet's balancing problem is classified as a GALBP, due to multiple product variants being produced, variable cycle times of workstations, variable processing times of tasks and unequal distribution of resources.

In Section 4.2 stated the solution approach for Trioliet's balancing problem. The solution approach has two steps:

1. Initialize starting situations (initial solutions)
2. Improve the initial solutions (neighbor solutions)

In Section 4.3, we described the first step of the solution approach, where we discussed the solution method, constraints and the iteration procedure. The solution method describes that a task is randomly selected from a list of available tasks, formed based on precedence position. The selected task is assigned to the first available workstation. The initial solutions should satisfy zoning, precedence and maximum workload constraints.

In Section 4.4, we described the second step of the solution approach, where we discussed the solution method, constraints and the iteration procedure. The Hill Climbing method is used in a semi greedy way to explore neighbor solutions, where tasks are moved or swapped. The TU line encounters zoning and precedence constraints for assigning tasks to workstations.

In Section 4.5, we discussed the objectives to measure workload balance of the final solutions. Four objectives were chosen to assess the final solutions:

1. Maximize the minimum workload
2. Minimize the maximum workload
3. Minimize the difference between maximum and minimum workload
4. Minimize the mean absolute deviation (MAD)

In Section 4.6, we have established a detailed list of tasks with zoning constraints. Zoning constraints are used to refine the list of tasks where compatible (sub)tasks are combined into one task. Then, precedence constraints are generally established where it satisfies all product variants at the TU line.

Chapter 5 Results

Chapter 4 described the solution approach and this chapter presents the solutions for the balancing problem. Section 5.1 describes the execution and results of the initial solutions. Next, Section 5.2 describes the execution and the results of the improved initial solutions (neighbor solutions). Further, Section 5.3. discusses the proposed solution. Then, Section 5.4 estimates the throughput time improvement and Section 5.5 concludes this chapter.

5.1. Initial solutions

Subsection 5.1.1 describes the execution of the first step of the solution approach and the results of the initial solutions in Subsection 5.1.2.

5.1.1. Execution of step ‘generating initial solutions’

The first step of the solution approach generates initial solutions and Subsection 4.3.3 described an iterative procedure where a task is selected and then assigned to a workstation. We execute this iteration procedure by hand in Excel. The first step in the iteration, 'select a task' is done in Excel where we use a short macro to choose a task randomly from a list of available tasks. The list of available tasks is made based on precedence relations, where we ‘release’ at most 3 tasks at each iteration, because each task in the precedence diagram has at most 3 neighboring tasks. The button 'Iterate 1x' creates a list of available tasks and selects one random task from it (Figure 16, box 6). Appendix B shows the VBA code used for creating a list of available tasks and selecting a task randomly from this list.

The second step in the iteration procedure, 'assign a task’, we manually assign a task to the first possible workstation in Excel, according to the precedence constraints and the maximum workload of a workstation. We stop the iteration procedure when all tasks are assigned and we can use the button 'begin again' (box 6 in Figure 16) to reset the task distribution, which removes all tasks in box 1 (in Excel).

Figure 16 shows a screenshot of Excel and consists of six parts (boxes):

1. Workload column: column for assigning tasks to workstations
2. Input column: column with inputs like workload of tasks, task descriptions and task numbers
3. Columns ‘al gedaan?’ and ‘Mogelijk om te selecteren?’ are needed for generating a list of available tasks per iteration
4. Table for the bar graph
5. Bar graph shows the workload and task distribution
6. Button ‘Iterate 1x’ generates a list of available tasks and selects one task randomly and button ‘Begin again’ removes all the tasks from the workstations (empty column ‘workstation’ (box 1))

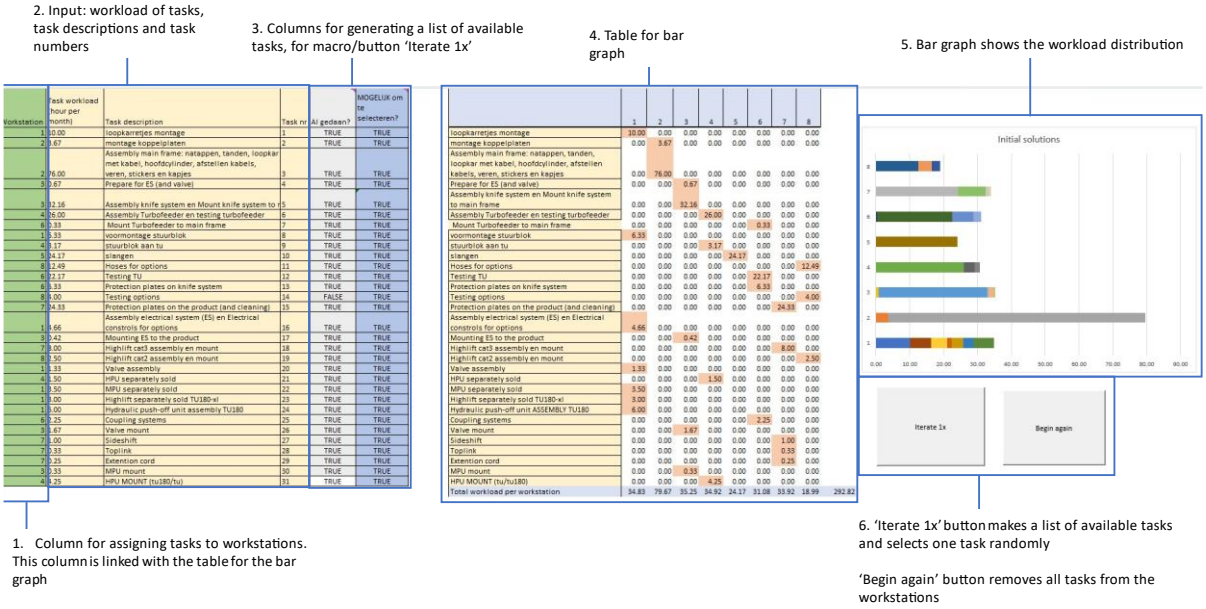


Figure 16: Screenshot of excel sheet for the first step of the solution approach

5.1.2. Results initial solutions

We have four initial solutions, where the first solution is the current task allocation at the TU line and the other task allocations are randomly drawn with the priority rule of 'precedence position'. Table 8 shows the workload distribution and the objective function values (MAD, minimum workload, maximum workload and difference between minimum and maximum) of the four initial solutions. The 'minimum workload' objective is marked green and the 'maximum workload' objective is marked red in the table. The 'MAD' objective and 'difference between min and max' objective have their own columns in the table.

- Initial solution 1
  Chapter 2 determined the current task and workload distribution at the TU line. The MAD objective value of the current TU line is 14.27 hours (per month). The minimum workload is 13.67 hours at workstation 1 and the maximum workload is 81.25 hours at workstation 2. The difference between min and max is 67.58 hours.

- Initial solution 2
  In initial solution 2, the MAD objective value is 10.76 hours. The minimum workload is 18.99 hours at workstation 8 and the maximum workload is 79.67 hours at workstation 2. The difference between min and max is 60.68 hours.

- Initial solution 3
  In initial solution 3, the MAD objective value is 14.49 hours. The minimum workload is 24.17 hours at workstation 5 and the maximum workload is 76.00 hours at workstation 2. The difference between min and max is 51.83 hours.

- Initial solution 4
  In initial solution 4, the MAD objective value is 9.85 hours. The minimum workload is 18.99 hours at workstation 1 and the maximum workload is 76.00 hours at workstation 2. The difference between min and max is 57.01 hours.

Workstations	1	2	3	4	5	6	7	8	MAD value (hours)	(Min - Max) value
Initial solution 1	13.67	81.25	32.17	26.00	35.67	30.67	24.33	49.07	14.27	67.58
Initial solution 2	34.83	79.67	35.25	34.92	24.17	31.08	33.92	18.99	10.76	60.68
Initial solution 3	26.16	76.00	24.33	32.16	24.17	28.83	26.00	55.16	14.49	51.83
Initial solution 4	35.66	76.00	36.00	34.08	27.33	28.83	35.92	18.99	9.85	57.01

Table 8: Workloads distribution and objective values (MAD, min, max, diff max-min) of the initial solutions

5.2. Neighboring solutions

Subsection 5.2.1 discusses the execution of generating neighbor solutions (the second step of the solution approach) and Subsection 5.2.2 shows the results of the neighboring solutions where workload balance is assessed.

5.2.1. Execution of the second step of the solution approach

The second step of the solution approach is to improve the initial solutions. Subsection 4.4.2 described an iterative procedure where a task is selected and then either moved to another workstation or swapped with another task. We have executed this iteration procedure by hand in Excel.

The first step in the iteration is to ‘choose a workstation (that still can be changed)’, where we start with the workstation with the highest workload to the second highest workstation. We choose a workstation based on the information of box 6.1. (workload of workstations) in Figure 17.

The second step in the iteration is 'select a task', we select the task from highest to lowest workload at the chosen workstation. We select a task based on the information of box 2 (workload of tasks) in Figure 17.

The third step is to find neighbor solutions with the operators: move or swap. We have executed this by changing the values of box 1 (column ‘workstations’) in Figure 17. The neighbor solutions are accepted when the precedence constraints are satisfied, and objective function has improved. The objective value and its parameters are shown in box 5 in Figure 17.

Figure 17 consists of 6 parts:

1. Workload column: column for assigning tasks to workstations
2. Input column: column with inputs like workload of tasks, task descriptions and task numbers
3. Table for the bar graph
4. Bar graph shows the workload and task distribution
5. Calculation of the objective function ‘average difference from the mean workload’
6. Calculation of the workload of workstations (6.1.) and the calculation of the difference between the workload and the mean workload of a workstation (6.2).

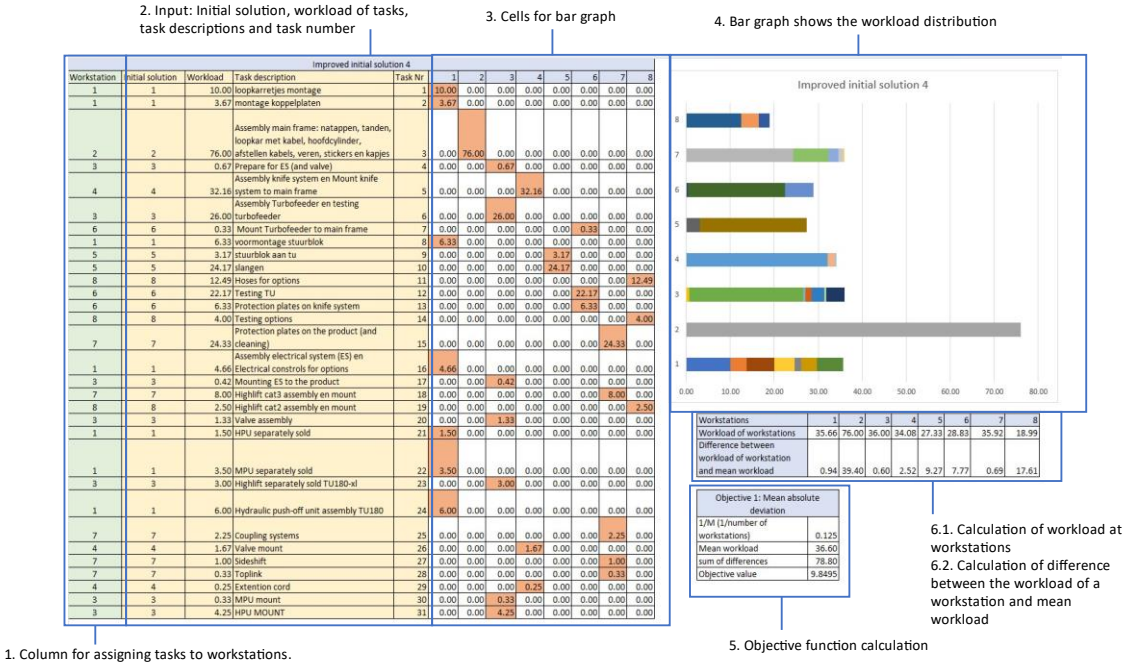


Figure 17: Screenshot of excel sheet for the second step of the solution approach (example of initial solution 4)

5.2.2. Results of neighboring solutions

This subsection discusses two parts. The first part compares the neighbor solutions with the initial solutions. The second part compares the neighbor solutions with the current situation and chooses the best solution for Trioliet's problem.

Initial solutions vs neighbor solutions

This part provides context on how much the workload balance has improved between the initial and neighbor solutions with our solution approach. In Chapter 4, we have chosen four objective measures to evaluate neighbor solutions: Mean Absolute Deviation (MAD), minimum workload (min), maximum workload (max) and difference between minimum and maximum workload. We are going to use these four objectives to compare the initial and neighbor solutions. Table 9 shows the comparison between the initial and neighbor values of the objective functions and the percentual change of the objective functions. Three conclusions are drawn from Table 9:

1. The use of a constructive heuristic does not always lead to a better objective function value (compared to the current situation). For instance, when we look at the initial MAD value of solution 3: we observe that initial solution 3 has a worse MAD value of 14.49 compared to 14.27 of the initial solution 1 (marked in blue in Table 9).

2. The use of the constructive heuristic does not necessarily lead to the best objective function value (compared to the current situation). For instance, when we look at the initial 'diff min-max' value of the solutions 1, 3 and 4: we observe that initial solution 4 has a better value of 57.01 than 67.58 of initial solution 1 but initial solution 3 has the best objective value of 51.83 (marked in purple in Table 9).

3. The use of improvement heuristic does not necessarily lead to the best objective function value (compared to the current situation), because you can start from a poor initial objective value. For instance, initial solution 1 starts with the worst value for 'Min' of 13.67 and has the greatest improvement of 31%, but it is not the best objective value (marked in green in Table 9).

	Solution 1			Solution 2			Solution 3			Solution 4		
	Initial	Neighbor	% Delta	Initial	Neighbor	% Delta	Initial	Neighbor	% Delta	Initial	Neighbor	% Delta
MAD	14.27	9.85	-45%	10.76	9.85	-9%	14.49	9.85	-47%	9.85	9.85	0%
Min	13.67	19.67	31%	18.99	22.49	16%	24.17	28.50	15%	18.99	18.99	0%
Max	81.25	76.00	-7%	79.67	76.00	-5%	76.00	76.00	0%	76.00	76.00	0%
Diff min-max	67.58	56.33	-20%	60.68	53.50	-13%	51.83	47.50	-9%	57.01	57.01	0%

Table 9: Objective function values between initial and neighbor solutions

To conclude, the best solution is found by using both constructive and improvement heuristics because it allows starting with the 'best' initial values and applying improvements from there, where it is likely to reach or come close to a local optimum. Table 9 does not snow that the solutions have reached their local optimum.

Neighbor solutions vs current situation

This part compares the neighbor solutions with the current situation on 'workload balance'. Appendix C.1 shows the workload distribution of the neighbor solutions. The neighbor solutions are assessed with the same four objectives functions that were used to compare initial and neighbor solutions: MAD, minimum workload, maximum workload and difference between maximum and minimum workload. Table 10 shows the objective values of the different neighbor solutions and the current situation.

	Current situation	Solution 1	Solution 2	Solution 3	Solution 4
MAD	14.27	9.85	9.85	9.85	9.85
Min	13.67	19.67	22.49	28.50	18.99
Max	81.25	76.00	76.00	76.00	76.00
Diff min-max	67.58	56.33	53.50	47.50	57.01

Table 10: Objective values of neighbor solutions and the current situation

- MAD objective

The MAD objective aims to be minimized and all neighbor solutions performed better than the current situation. Each solution ended up in the minimum of 9.85 hours with some negligible differences (Table 10). This happened because of the nature of the chosen objective function, as the MAD is not always able to minimize peaks in workload, because the workload changes are cancelled out. (Vanheusen, 2020) In the case of Trioliet's solutions, we observe that the overall mean workload lies higher than the workload of the workstations, which results in the MAD objective cannot be improved anymore.

- Minimum workload

This objective function aims to maximize the minimum workload, so that it lies closer to the mean workload of the production line (36.6 hours per month). The third solution performed best with the maximum value of 28.5 hours (per month), which is an increase of 14.83 hours compared to the current situation (marked green in Table 10).

- Maximum workload

This objective function aims to minimize the maximum workload, so that it lies closer to the mean workload of the production line. The objective could be minimally improved since the workstation with the highest workload could be minimally improved because of zoning and precedence constraints. All solutions ended up in a maximum workload of 76 hours (per month) (marked green in Table 10).

- Difference between maximum and minimum workload

This objective function aims to minimize the difference between maximum and minimum workload, and it has improved compared to the current situation. The third solution performs best with a minimum value of 47.5 hours (per month) (marked green in Table 10).

To conclude, Table 10 shows that neighbor solution 3 performs the best on the objective measures. Therefore, solution 3 is considered the proposed solution for Trioliet's balancing problem.

5.3. Proposed solution

Section 5.2 concluded that solution 3 is the proposed solution for Trioliet's balancing problem. This subsection has 2 parts. Subsection 5.3.1 discusses the task distribution of the proposed solution and Subsection 5.3.2 discusses the workload distribution of the proposed solution.

5.3.1. Task distribution of proposed solution

The proposed solution shows that 18 tasks have been reassigned to different workstations compared to the current situation. Appendix C.2 shows the full task distribution of the current situation and the proposed. Figure 18 shows the task distribution of the current and the proposed solution and four conclusions can be drawn:

1. Tasks related to independent assemblies were reassigned, such as the assembly of the control block and of multiple options, because they are easy to move as they are not dependent on other tasks (marked green in Figure 18). The effect is that some components are assembled at a certain workstation, but they are mounted to the product at another workstation.

2. Multiple tasks related to the mounting of options or other parts have been moved towards earlier workstations in the production process. For example, the valve and control block can be mounted to the product earlier (without attaching hoses) (marked blue in Figure 18).

3. The task 'hoses for options' is fully carried out at the last workstation due to the precedence relations (marked orange in Figure 18)

4. The task 'testing of options' is fully carried out at the last station because it is the last task in the precedence diagram (marked yellow in Figure 18).

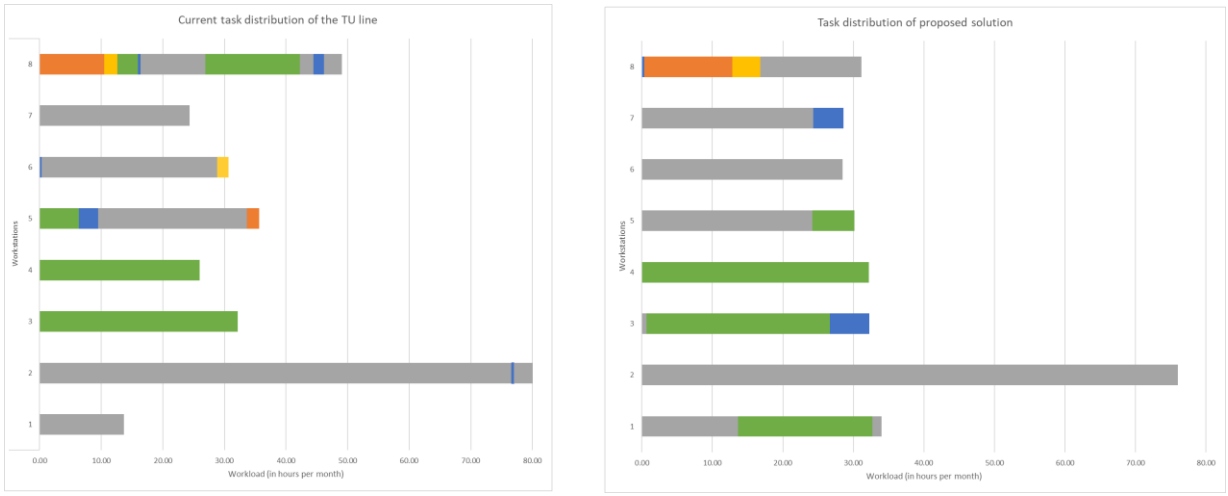


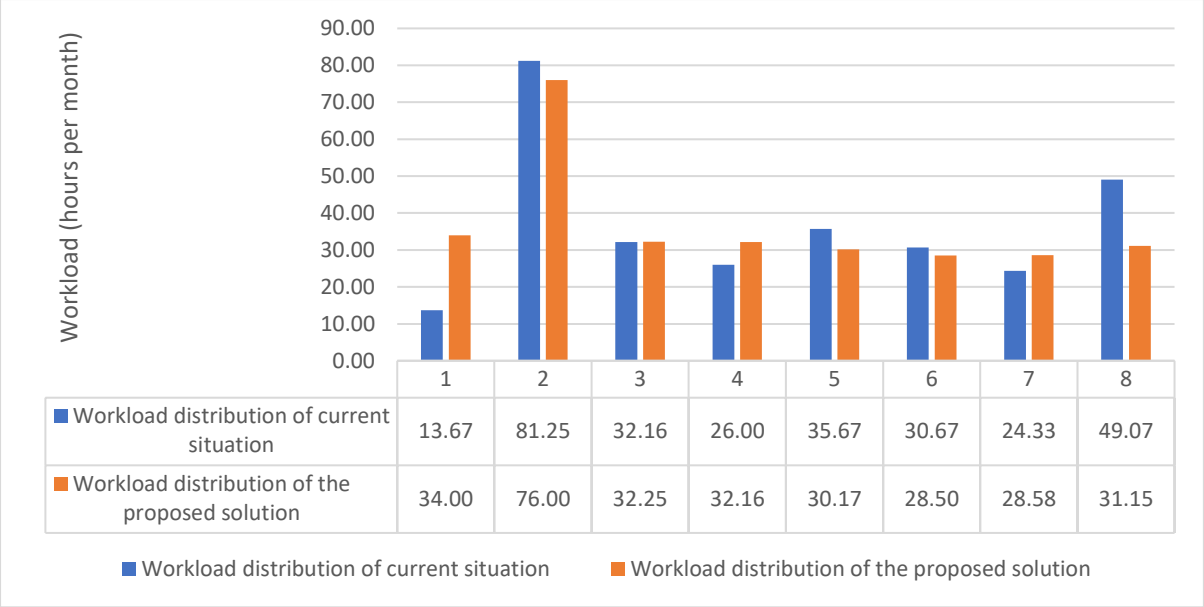
Figure 18: Task distribution of the current situation and of the proposed solution

5.3.2. Workload distribution of proposed solution

The proposed solution has improved the workload distribution at the TU line, when compared to the current situation. We make four conclusions:

1. Workstations 2 and 6 have a reduction in workload of less than 10%
2. Workstations 4 and 7 have an increase in workload between 10% and 20%
3. Workstations 5 and 8 have a reduction in workload more than 15%
4. Workstation 1 has an increase larger than 20%

Figure 19 illustrates the workload distribution for both the current situation and the proposed solution.



	1	2	3	4	5	6	7	8
Workload distribution of current situation	13.67	81.25	32.16	26.00	35.67	30.67	24.33	49.07
Workload distribution of the proposed solution	34.00	76.00	32.25	32.16	30.17	28.50	28.58	31.15

Figure 19: Workload distribution of the current situation and the proposed solution

5.4. Estimation of throughput time improvement

Chapter 1 introduced the action problem 'high average throughput time per month at the TU line' and formulated the research question as 'how can Trioliet reduce the average throughput time with 10% by solving the workload imbalance at the TU line?'. Section 5.2 explored multiple balancing solutions for the TU line and determined the best performing solution on workload balance, which was the third solution. The third balancing solution is therefore the proposed solution. This section aims to translate the proposed solution back to 'throughput time'.

Subsection 5.4.1 defines throughput time and cycle time and explains how they are calculated in this research. Subsection 5.4.2 explains the Kingman relation that is used to calculate expected waiting time at workstations. Subsection 5.4.3 shows the results of the sensitivity analysis.

5.4.1. Throughput and cycle time (definition + calculation)

- Throughput time at a production line

Throughput time was already defined in Chapter 1 as 'the time from the start of the first to the completion of the final processing step'. This research calculates the average throughput time of the production line (per month) by summing the cycle time of all workstations (per month) (Equation 3).

$$\mathrm{Th} = \sum_{i=1}^{8} c_i \qquad (3)$$

- Cycle time at a workstation

J.Hopp (2008) states that the average cycle time at a workstation is made up from the following components:

1. Process time
2. Queue time (waiting time)
3. Move time
4. Setup time
5. Wait-to-batch time, wait-in-batch time, wait-to-match time

Process time is 'the time jobs are being worked on at the station' (J.Hopp, 2008), which is called workload in this research. Chapter 2 already defined workload with the definition from Dinler (2021). The

workload of workstations of the current situation at the TU line is known from Chapter 2 and Section 5.2 calculated the workload of workstations of the balancing solutions.

Queue time is 'the time spent waiting for processing at the workstation or moved to the next workstation'. (J.Hopp, 2008) In this research, we refer to queue time as waiting time. The waiting time is not known in this research; however, it can be estimated with the Kingman relation. The Kingman relation is explained in Subsection 5.4.2.

Move time is 'the time jobs spend being moved from the previous workstation'. (J.Hopp, 2008) The move time at the TU line is expected to be low, as the used space for the TU line is quite small, therefore we do exclude the move time in this research.

Setup time is 'the time a job spends waiting for the station to be set up'. (J.Hopp, 2008) The TU line has relatively few setup times for its tasks, and the tasks that require setup time are mainly for positioning frames. The setup time is excluded in this research, because of two reasons. First, the setup times do not play a large role at the TU line. Second, taking setup times into account would take too much time in this research.

Wait to batch, wait in batch and wait to match time can be excluded in this research as the TU line does (for the most part) not work with batches of products across workstations. Also, considering batches would make this problem too complex, as these batches would vary in size.

To conclude, cycle time at a workstation is calculated by summing up the workload at a workstation (per month) and the expected waiting time at a workstation (per month) (Equation 4).

$$\begin{aligned} C_i = {} & \text{Workload at workstation i} \\ & + \text{Expected waiting time at workstation i} \end{aligned} \quad (4)$$

5.4.2. Kingman relation

This subsection is divided into three parts. The first part defines the Kingman function. The second part explains the relationship between the variables of the Kingman function. The third part shows an example of how the Kingman relation is used in the research.

Kingman function

The Kingman relation estimates the expected waiting time at a workstation ($E(W)$) which is needed in this research to calculate the cycle time of a workstation (per month). $E(W)$ is dependent on 3 factors: variability (V), utilization ratio (U) and lead time (T) (Equation 5).

$$E(W) = V * U * T \quad (5)$$

- The variability of a workstation (V) consists out of the variability of the service and arrival rates at a workstation. High variability can lead to higher queue lengths and thus leads to higher expected waiting times. In this research, we assume that the variability remains constant when the workload at a workstation changes (as the task distribution at the TU line changes). Since the exact value of V is unknown, we set $V = 1$ for all workstations, based on the assumption that variability is low at the workstations.

- U is a ratio of the workstation utilization (ρ) (Equation 6). ρ is calculated by dividing the actual workload at a workstation in a certain period by the available capacity at a workstation in a certain period (Equation 7). This research uses the workload distribution of the proposed solution and the current workload distribution from Chapter 1 as input for the 'actual workload'. The available capacity of a workstation is not known because of two reasons: the number of operators at the TU line changes and it is not known how operators are distributed among the workstations. Therefore, we conduct a sensitivity analysis in Subsection 5.4.3.

$$U = \frac{\rho}{(1-\rho)} \quad (6)$$

$$\rho = \frac{actual\ workload\ at\ a\ workstation\ in\ a\ certain\ period}{available\ capacity\ at\ a\ workstation\ in\ a\ certain\ period} \quad (7)$$

- The lead time (T) is 'the average processing time in the workstation or the assembly line'. (J.Hopp, 2008) This research defines the lead time of a workstation as 'the average workload of a workstation'. The average workload of a workstation over a specific period is unknown in this research, therefore, we assume that the workloads collected in Chapters 2 and 4 represent the average workload of a workstation.

Relation between workstation ρ and E(W) at a workstation.

In the first part of this subsection, we have defined the variables of Kingman: variability (V), utilization ratio (U) and lead time (T). We also made some assumptions:

- V = 1 and that it remains constant when the workload at a workstation changes.
- T is equal to the 'actual workload' of the workstation

With this information, we understand how the variables of Kingman relationship are used in our research. In our research, only changes in the variables U (more specifically: utilization (ρ)) and T, cause a change in $E(W)$. We describe the relationship between ρ and $E(W)$.

The workstation utilization (ρ) and the expected waiting time of a workstation ($E(W)$) have an exponential relation, because of the utilization ratio $U = \frac{\rho}{(1-\rho)}$. The lead time ($T$) have a linear relation with $E(W)$. Figure 20 shows the relationship between the ρ, T and $E(W)$. The x-axis shows the ρ value of a workstation and the y-axis shows $E(W)$ value of a workstation. In this example, we assume V = 1 and take T = 81.25 (workload of workstation 2). Appendix D.1 shows the full calculation used for Figure 20. We draw two conclusions regarding the relation between ρ and $E(W)$:

1. The higher ρ at a workstation, the higher $E(W)$ is.
2. The higher the ρ at a workstation, the greater the change in $E(W)$ is when ρ changes. This holds for when the ρ changes upwards and downwards. For example, if ρ changes from 0.9 to 0.8, we see a reduction of $E(W)$ from 731.25 to 325 hours (-406.25 hours). However, if ρ changes from 0.5 to 0.4, we only see a change in $E(W)$ from 81.25 to 54.17 hours (-27.08 hours).

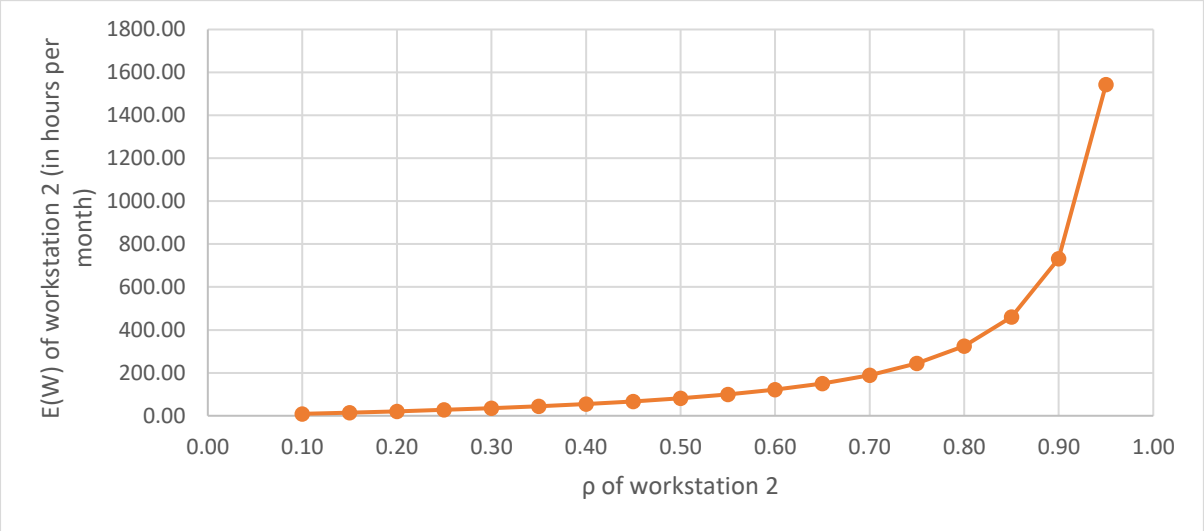


Figure 20: Relation between ρ and E(W)

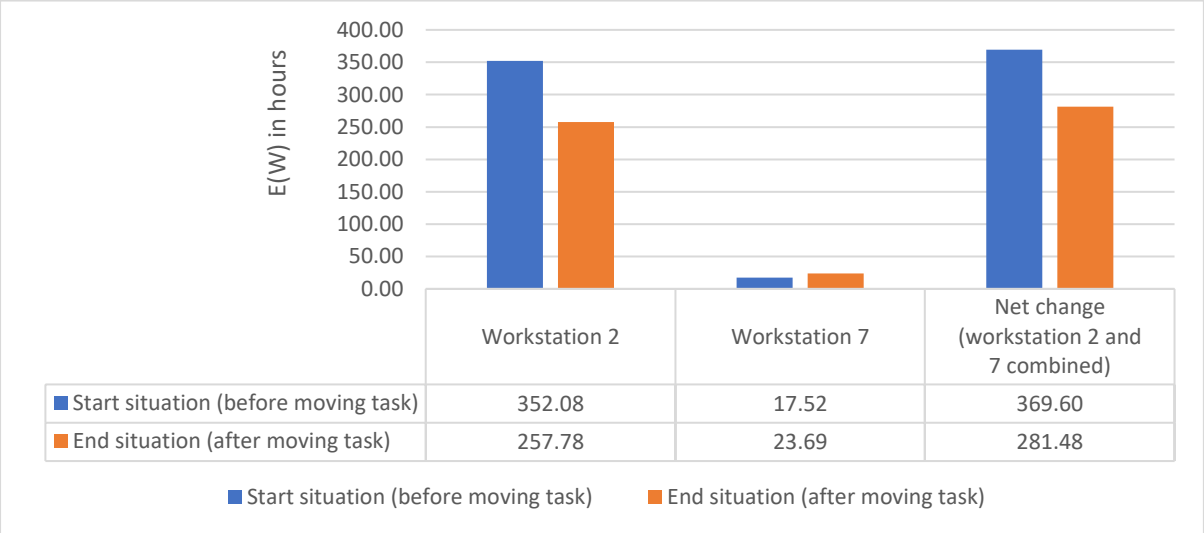
Execution of Kingman relation; example of moving one task

In the previous part of this subsection, we discussed the relation between ρ and $E(W)$ and concluded that changing a highly utilized workstation in terms of ρ results in a significant change in $E(W)$. Also, changing a low utilized workstation in terms of ρ results in a small change in $E(W)$. With this relationship, our research aims to reduce the overall $E(W)$ of all workstations by significantly decreasing $E(W)$ at highly utilized workstations and slightly increasing $E(W)$ at low utilized workstations.

We explain our objective with an example where we move the task "HPU mount" (workload = 4.25 hours per month) from workstation 2 to 7. This example chooses a start utilization of 81% for workstation 2, assumes that $V = 1$ (and stays constant for both workstations when the task is moved) and assumes that the available capacity of both workstations is equal.

Workstation 2 starts with ρ = 81% (before moving the task) and the ρ is lowered to 77% (after moving the task), this results in an $E(W)$ decrease of 94.3 hours at workstation 2. Workstation 7 starts with ρ = 34% (before moving the task) and the ρ is increased to 38% (after moving the task), this results in an $E(W)$ increase of 6.18 hours at workstation 3. Conclusively, the overall $E(W)$ of both workstations has decreased by 94.3 - 6.18 = 88.12 hours (after moving the task).

Figure 21 shows the start situation and end situation of $E(W)$ for workstation 2, 7 and combined, and Appendix D.2 shows the full calculation of this example.



	Workstation 2	Workstation 7	Net change (workstation 2 and 7 combined)
Start situation (before moving task)	352.08	17.52	369.60
End situation (after moving task)	257.78	23.69	281.48

Figure 21: Kingman example: start/end E(W) values of workstation 2, 7 and combined

5.4.3. Sensitivity analysis

Subsection 5.4.2 discussed the Kingman relation where $E(W)$ is calculated with the utilization, variability and lead time of a workstation. This research does not know the utilization at the workstations; therefore, we conduct a sensitivity analysis, which 'determines how different values of an independent variable affect a particular dependent variable under a given set of assumptions', according to Kenton (2023). In this research, the dependent variable is ρ and the independent variable is $E(W)$.

The sensitivity analysis conducts the example 'Execution of Kingman relation' from Subsection 5.4.3 on a larger scale, where the same idea of reducing the overall $E(W)$ of all workstations is applied. The sensitivity analysis calculates multiple cases of the percentual reduction of the throughput time of the proposed solution (in comparison with the current situation), where the start utilization of workstation 2 is varied between 81 and 90%. We assume that the available capacity of all workstations is the same, thus the workstation utilization of other workstations can be easily calculated. Further, we assume that $V = 1$, which means that variability is low.

This subsection is divided into two parts. Part 1 explains the range of utilization chosen for the sensitivity analysis. Part 2 discusses the results of the sensitivity analysis.

Explanation utilization range

The workstation utilization (ρ) ranges between 81% and 90% for workstation 2, which is equivalent to 90 to 100 hours per month available capacity (on average) per workstation. The range of average available capacity for a workstation must be between 82 and 160 hours per month. The lower bound is 82 hours for workstation 2 as we assume that utilization cannot be higher than 100%, but we take a lower bound of 90 hours (arbitrarily chosen). The upper bound is 100 hours, which is 8 hours per day * 5 workers * 5 days per week * 4 weeks per month / 8 workstations = 100 hours available on average per workstation per month.

Sensitivity analysis results

Figure 22 shows the results of the sensitivity analysis, where we observe that the proposed solution can reduce the throughput time (per month) by 14.56 to 29.21%, depending on the workstation utilization of workstation 2.

The throughput time improvement is higher when ρ of workstation 2 is higher (Figure 22). A higher ρ for workstation 2 means that there is less available capacity, thus less operators at the TU line.

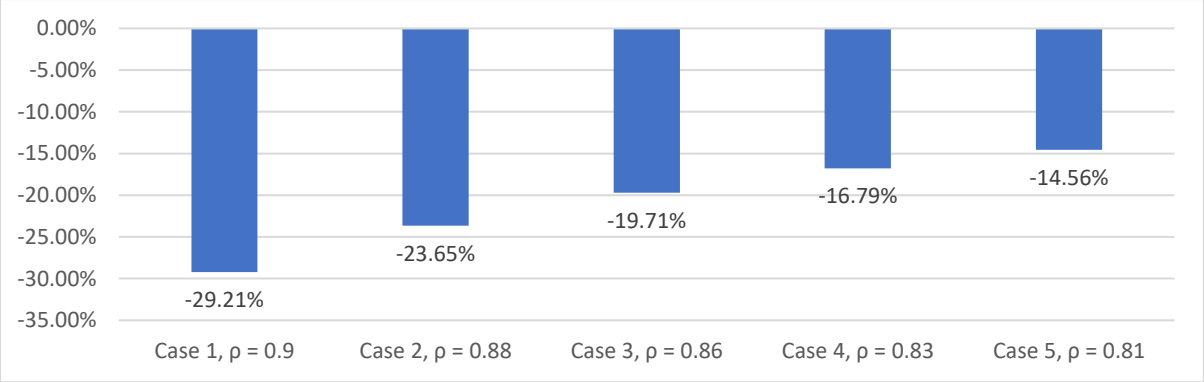


Figure 22: Throughput time reduction for the different ρ cases for the proposed solution (sol3)

5.5. Conclusion

Chapter 5 executes the solution approach from Chapter 4.

Section 5.1 explains the execution of the initialization of the starting situations and shows the results of these initial solutions. The execution of the iteration procedure is twofold: tasks were selected by a small macro in Excel and tasks were assigned to workstations by hand in Excel.

Section 5.2 explains the execution of improving the initial solutions (neighbor solutions) and evaluates the performance of the neighbor solutions. The iterative procedure is executed by hand in Excel (selecting and assigning tasks to workstations) and the calculations of the workload of workstations and the objective function are executed in Excel. Then, we compared the initial and neighbor solutions with each other and found out that the improvement heuristic made a real difference for improving the 'workload balance' of the solutions. Last, the section concluded that the third workload balancing solution is the best solution for solving Trioliet's problem, because it performed best on the objective measures 'minimum workload' and 'difference between min and max'.

Section 5.3 provides context on the proposed solution, explaining the task and workload distribution. The task distribution of the proposed solution shows that primarily independent assemblies have been moved. Additionally, some tasks related to the mounting of options/components are moved to earlier workstations in the production process. Lastly, the tasks 'hoses for options' and 'testing of options' are fully positioned at the last station due to precedence relations. The workload distribution shows four changes:

1. The workload of workstations 2 and 6 has decreased by less than 10%.
2. The workload of workstations 5 and 8 has decreased by more than 15%.
3. Workload of workstations 4 and 7 have increased between 10% and 20%
4. The workload of workstation 1 has increased by more than 20%.

Section 5.4 estimates the throughput time improvement for the proposed problem and this section is divided into three subsections.

Subsection 5.4.1 explains how throughput time, cycle time and waiting times are calculated. Throughput time consists of the cycle times of all workstations. Cycle time at a workstation consists of process time at a workstation and waiting time at a workstation. Waiting time is estimated with the Kingman relation, where variability, utilization and lead time are multiplied.

Subsection 5.4.2 explains the Kingman relation, where the variables and its relations are explained. The Kingman relation states that $E(W) = V * U * T$, which means that expected waiting time at a workstation can be calculated by multiplying the variability, utilization ratio and lead time of a workstation. The utilization has an exponential relationship with $E(W)$ where a higher utilized workstation has an exponentially higher $E(W)$. Variability and lead time have a linear relationship with EW. In our research, we assume that $V = 1$ (and remains constant after a change in task distribution), so only U * T has an effect on $E(W)$. The research wants to reduce the overall $E(W)$ of all workstations, where we decrease the $E(W)$ significantly at high utilized workstations and increase $E(W)$ a little bit at low utilized workstations. We show this principle and how we use the Kingman relation in our research with an example, where we shift a task between workstation 2 and 7.

Subsection 5.4.3 conducts a sensitivity analysis, where multiple throughput time reductions for the proposed problem are calculated, by varying the workstation utilization of workstation 2. The workstation utilization is varied between 81 to 90% and assumes that the available capacity is the same for all workstations, thus the utilization of other workstations can be calculated. Conclusively, the proposed solution results in a throughput time reduction between 14 and 29%

Chapter 6 Conclusions & recommendations

Chapter 6 concludes the research. Section 6.1 discusses the research main findings and Section 6.2 discusses the research limitations and potential future research for the company.

6.1. Research main findings & feasibility & recommendations

Subsection 6.1.1 discusses the main findings of the research. Subsection 6.1.2 discusses implementation issues. Then, Subsection 6.1.3 gives a recommendation.

6.1.1. Main findings

The research objective is to improve the throughput time at the TU line and the main cause is that the TU line experiences a workload imbalance across workstations. The research focuses on changing the task distribution at the TU line to improve the workload balance.

We began by determining the current task and workload distribution and found that tasks and workload were relatively well distributed across the workstations. However, workstation 2 has the highest workload per month because it handles the largest task of the process. Additionally, workstation 8 has a higher workload due to performing many tasks related to assembly, mounting, and testing of options. Otherwise, the TU line is fairly balanced in terms of workload.

Next, we formulated a solution approach, where we used various heuristics to solve the problem. We start by creating initial solutions where we fill empty workload distributions with tasks. Then, we improve these initial solutions by shifting or swapping tasks. In total, we formulated four different solutions using our method and selected the best solution based on four objective functions: Mean Absolute Deviation (MAD), minimum workload, maximum workload, and the difference between minimum and maximum workload. We concluded that solution 3 is the best solution for Trioliet based on these objective functions, thus it is the proposed solution of this research.

The proposed solution improved the workload balance by reducing the workload of workstations 2 and 8 and increasing the workload around these stations. The task distribution of the proposed solution resulted in 18 tasks being shifted to other workstations, compared to the current task distribution. Four conclusions can be made:

1. Tasks related to independent assemblies were reassigned, for example assembly of control block and assembly of multiple options were reassigned.
2. Multiple tasks related to the mounting of options or other parts have been moved towards earlier workstations of the production process. For example, the valve and control block can be mounted to the product earlier (without attaching hoses)
3. The task ‘hoses for options’ is fully carried out at the last workstation due to the precedence relations.
4. The task ‘testing of options’ is fully carried out at the last workstation due to the precedence relations.

We conducted a sensitivity analysis for the proposed solution to estimate the percentual throughput time improvement. For this, we used the Kingman formula, which indicates that the expected waiting time of a workstation is the multiplication of utilization, variability and lead time of a workstation. We varied the utilization of workstation 2 between 81 and 90% and we made 3 assumptions: available workload of all stations is equally distributed, variability is equal to 1 and the variability remained constant (when changing the task distribution). This resulted in a throughput time improvement of 14 to 29 percent for the proposed solution.

6.1.2. Limitation for implementation & recommendation

This subsection discusses the limitations of the proposed solution and gives a recommendation for the company.

Limitation

The proposed solution has one potential implementation issue, which is that there is limited available space in terms of layout, which could mean that materials, inventory, and worktables possibly cannot be relocated.

Recommendation

This subsection answers the question: 'is it worth implementing the proposed solution?'. Two points are discussed:

Firstly, the proposed solution leads to distributing more complex tasks across workstations, which results in more line operators having to work on these complex tasks. Only the head operator can handle complex tasks, so more training is required for the other line operators.

Secondly, the throughput time improvement is estimated with 5 cases by varying the available capacities of workstations. This research we cannot precisely determine what the throughput time reduction is because precise available capacities of workstations are not known (utilization is not known).

To conclude, the proposed solution seems worthwhile to implement because the throughput time can be reduced by at least 14%. However, the company must assess its feasibility considering the lack of available space and determine if it is worth training the line operators so they can also perform the complex tasks.

6.2. Limitations of the research & future research at the TU line at Trioliet

In this section, we discuss future research that Trioliet can conduct for the TU line. Subsection 6.2.1 discusses the research limitations. Next, Subsection 6.2.2 discusses how this research could have made different decisions and what Trioliet can do for future research into the task allocation at the TU line. Last, Subsection 6.2.3 discusses potential future research topics for the TU line.

6.2.1. Limitations of the research

This research has encountered limitations in data gathering, which should be mentioned for the evaluation of the research.

- Data availability

This research has not measured the throughput time, due to time constraints and since it was not available.

- Qualitative data gathering

The process time of tasks are collected qualitatively through interviews, which could be sensitive to the judgement or bias of line operators. Therefore, the data could be untrustworthy.
Additionally, in the research, we found that the monthly workload is relatively low. This might be because the collected process times of tasks were undervalued or that product demand (of 2022) is not fully included.

6.2.2. Future research into task allocation at the TU line

This research on task allocation made certain decisions and we discuss how they could have been made differently in future research. There are multiple improvements that can be considered.

- Approach regarding workload balance

Workload balancing can be achieved in two ways: distributing workload across workstations or increasing the capacity at bottleneck workstations. Our research decided to solve the workload distribution problem, where tasks are tactically distributed. In the future, Trioliet could look at the distribution of available capacity across the workstations, for example, the distribution of available operating hours across workstations.

- Exact method vs heuristic

In our research, we have chosen to use heuristics, instead of exact methods. Exact methods calculate every possible outcome and heuristics rely more on a practical approach. With exact methods, we could find an optimal solution, however these methods are complex and time consuming. We do not recommend Trioliet, to use exact methods in future research because its added value is minimal and the feasibility is low due to a lack of knowledge, computational power and available time.

- Local optimum vs global optimum

Our research has chosen for an improvement heuristic that would end up in a local optimum. For future research, we can choose to use an improvement heuristic that avoids a local optimum (ends up in a global optimum), to explore a larger solution space, thus finding better solutions. Examples are simulated annealing and tabu search.

- Number of initial solutions & local optimum

In our solution approach, we chose to generate four initial solutions, which is an arbitrary chosen number of initial solutions. We know that our used improvement heuristic converges to a local optimum and that it would not add extra value to generate more initial solutions when the local optimum is already reached. However, in our research, we do not know if we have found the local optimum with the four initial solutions. Due to practical reasons, like manually conducting the improvement heuristic in Excel, we have chosen not to pursue more initial solutions. In future research, if the heuristics were coded in Excel, a larger set of initial solutions can be generated, thus finding the local optimum is possible.

- The choice in objective function

In this research, we used an objective function to choose the best neighbor solution during iterations and when choosing the best balancing solution after iterating. We reflect on both aspects.

1. Objective function for choosing the best solution during the iteration procedure

   We found out that the objective function 'MAD' played a large role in how far we could improve the workload balance, because the MAD value did not change at some point as all workstations (that still could be changed in workload) lied under the mean workload of all workstations. This could mean that we found a relatively 'bad' local optimum for our solutions (it is not known). In future research, a better objective function could be chosen for during the iterations, for example the difference between minimum and maximum workload.

2. Throughput time as objective function for choosing the best solution (after iterating)

   In our research, we did not use 'throughput time' for choosing the best solution, because it was difficult to compare different solutions, as there were many different possible outcomes for each solution due to that utilization and variability at workstations were unknown. Throughput time can be used, in future research, to choose the best solution when the starting utilization and variability of the workstations are known. However, multiple assumptions should be made to simplify the problem so that each solution has one outcome:

   - Available capacity does not change at the workstations (when changing the task distribution)
   - Variability at the workstations do not change (when changing the task distribution)

- Precedence constraints

This research used general precedence constraints without considering individual product variants (directly), because there were too many different products, which makes the problem too complex. To give an impression of how many different products were made in 2022, the TU line produced 52 different products. The decision to generalize precedence relations (for all product variants) resulted in a smaller solution set of possible task allocations. In future research, the company can maybe specify different precedence relations for different product variants to explore a larger solution set.

- Number of workstations

This research has not looked at the number of workstations, since the action problem was focused on improving the cycle time of workstations. In future research, the company could start research into minimizing the number of workstations.

6.2.3. Future research into other research topics at the TU line

Chapter 1 introduced the research as that the 'average throughput time at the TU line should be improved' and it translated this action problem into a workload balancing problem that is focused on changing the task allocation at the TU line. Chapter 2 also states other problems at the TU line and in future research, Trioliet can explore research into other topics for improving the TU line.

- Distribution of line operators

Chapter 2 stated that the TU line has a shortage of line operators at the TU line and therefore it can start research into the distribution of line operators across the workstations for a specific product demand.

- Supply issues

Chapter 2 stated that the TU line encounters supply issues, and this often results in a workstation being idle. This can be researched in the future.

- Production scheduling

In the future, the company can decide to start research into the production scheduling of products at the TU line, aligned with corresponding product demand. Currently, standard products are prioritized on the line for quicker processing, which appears to be a good strategy at first glance, but this research topic can be explored.

- Layout

The layout of the production line can be researched, to find an optimum layout where, for instance, walking distances could be reduced.

- Education of line operators

From my observation and informal interviews, it is found that the skills and abilities of the individual line operators are different. This means that not all workstations can be utilized by every line operator. The company can start research to reduce this gap of knowledge on the production line.

Bibliography

Ağpak, K. (2011). *Two-sided U-type assembly line balancing problem.* Gaziante: International Journal of Production Research.

Alghazi, A. A. (2017). Balancing and Sequencing of Mixed Model Assembly lines. In A. A. Alghazi, *Balancing and Sequencing of Mixed Model Assembly lines* (pp. 25-25). Clemson: Tigerprints.

Antoine, M. (2016). *Iterated Local Search for dynamic assembly line rebalancing problem.* Nancy: Elsevier.

Becker, C. (2004). *A survey on problems and methods in generalized assembly line balancing.* Jena: ScienceDirect.

Boysen, N. (2007). *Assembly line balancing: Which model to use when?* Hamburg: Elsevier.

Boysen, N. (2021). *Assembly line balancing: What happened in the last fifteen years?* Jena: Elsevier.

Breginski, R. (2013). *ASSEMBLY LINE BALANCING USING EIGHT HEURISTICS.* Paraná: ResearchGate.

Çelik, M. T. (2023). *Solution of the assembly line balancing problem using the rank positional weight method and Kilbridge and Wester heuristics method: An application in the cable industry.* Bursa: Elsevier.

Dinler, D. (2021). *Exact solution approaches for the workload smoothing in assembly lines.* Ankara: ScienceDirect.

Dung, H. Q. (2019). *The Heuristic Methods for Assembly Line Balancing Problem: A Case of Vietnam Garment Industry.* Hue: International journal for research in applied science & engineering technology (IJRASET).

Fathi, M. (2017). *An optimization model for balancing assembly lines with stochastic task times and zoning constraints.* Skövde: IEEE Access.

Glover, F. (2015, 04 25). *Metaheuristics*. Retrieved from Scholarpedia: http://www.scholarpedia.org/article/Metaheuristics

Huang, H. C. (2006). *The workload balancing problem at air cargo terminals.* Singapore: Springer.

J.Hopp, W. (2008). Cycle time. In W. J.Hopp, *Factory physics* (p. 327). Michigan: University of Michigan.

Kellegöz, T. (2012). *An efficient branch and bound algorithm for assembly line balancing problems with parallel multi-manned workstations.* Ankara: Elsevier.

Kenton, W. (2023, 12 19). *Sensitivity Analysis Definition*. Retrieved from Investopedia: https://www.investopedia.com/terms/s/sensitivityanalysis.asp

Kiran. (2019). Sequencing and line balancing. In Kiran, *Production Planning and Control* (pp. 351-351). Oxford: Elsevier.

KUMAR, N. (2001). *Comparing the effectiveness of workload balancing objectives in FMS loading.* Taylor & Francis.

Lapierre, S. D. (2006). *Balancing assembly lines with tabu search.* Montreál: Elsevier.

Li, Z. (2019). *Iterated local search method and mathematical model for sequence-dependent U-shaped disassembly line balancing problem.* Wuhan: Elsevier.

Lindfield, G. (2019). Chapter 9 - Optimization Methods. In G. Lindfield, *Numerical methods* (pp. 433-483). Birmingham: Academic press.

Nguyen, T.-H. (2013). *Variable neighborhood search for the workload balancing problem.* Lancaster: Elsevier.

Niemueller, T. (2017). Chapter 13 - Benchmarking of Cyber-Physical Systems in Industrial Robotics: The RoboCup Logistics League as a CPS Benchmark Blueprint. In T. Niemueller, *Cyber-Physical Systems* (pp. 193-207). London : Academic Press.

Özcan, U. (2010). *Balancing parallel two-sided assembly lines.* Ankara: International Journal of Production Research.

Pastor, R. (2021). *LB-ALBP: the lexicographic bottleneck assembly.* Barcelona: University of Catalonia.

Roshani, A. (2015). *A simulated annealing approach for multi-manned assembly line balancing problem type II.* Genova, Italy: Elsevier.

Saif, U. (2014). *A survey on assembly lines and its types.* Heidelberg: Springer.

Scholl, A. (1996). *Simple Assembly Line Balancing - Heuristic Approaches.* Darmstadt: Kluwer.

Vanheusden, S. (2020). *Operational workload balancing in manual order picking.* UHasselt: Elsevier.

Wu, T. (2023). *Mixed-integer programming model and hybrid local search genetic algorithm for human-robot collaborative disassembly line balancing problem.* Chengdu: Taylor & Francis.

Xu, S. (2023). *A novel competitive exact approach to solve assembly line balancing problems based on lexicographic order of vectors.* Zhejiang: Elsevier.

Yang, C. (2011). *A multi-objective genetic algorithm for mixed-model assembly line rebalancing.* Xi'an: Elsevier.

Yuan, B. (2012). *A late acceptance hill-climbing algorithm for balancing two-sided.* New York: Springer.

Appendix A – Current situation TU line

A.1. Specific list of tasks & processing time and frequency of tasks

The table considers a multiplication factor 'x' for the frequency and task time for all tasks.

Task description	Workstation	Task time standard TU (min)	Task time TU180-XL (min)	Average frequency TU	Average frequency TU180-XL	Totals (per month in hours)
Pre-assembly walking charts	1	15.38	0.00	69.03	0.00	17.70
Pre-assembly coupling plates	1	5.65	0.00	69.03	0.00	6.50
Assembly main frame	2	120.00	120.00	56.64	10.62	134.52
Prepare for ES (and valve)	2	10.00	10.00	1.77	5.31	1.18
Assembly knife system	3	40.00	45.00	56.64	10.62	45.73
Mount knife system to main frame	3	10.00	10.00	56.64	10.62	11.21
Assembly Turbofeeder	4	600.00	0.00	3.54	0.00	35.40
Testing Turbofeeder	4	180.00	0.00	3.54	0.00	10.62
Mount Turbofeeder to main frame	6	10.00	0.00	3.54	0.00	0.59
Assembly hose system	5	50.00	60.00	56.64	12.39	59.59
Hoses for options	5	15.00	10.00	7.08	10.62	3.54
Hoses for options	8	15.00	10.00	12.39	12.39	5.16
Testing TU	6	35.00	35.00	56.64	10.62	39.24
Protection plates on knife system	6	10.00	10.00	56.64	10.62	11.21
Testing options	6	10.00	10.00	8.85	10.62	3.25
Testing options	8	10.00	10.00	12.39	10.62	3.84
Protection plates on the product (and cleaning)	7	40.00	30.00	56.64	10.62	43.07
Assembly electrical system (ES)	8	40.00	40.00	3.54	5.31	5.90
Mounting ES to the product	8	5.00	5.00	3.54	5.31	0.74
Highlift cat3 assembly	8	0.00	180.00	0.00	3.54	10.62
Highlift cat2 assembly	8	60.00	0.00	3.54	0.00	3.54
Valve assembly	8	20.00	20.00	1.77	5.31	2.36
HPU separately sold	8	15.00	0.00	10.62	0.00	2.66
MPU separately sold	8	15.00	0.00	24.78	0.00	6.20
Highlift separately sold	8	180.00	0.00	1.77	0.00	5.31
Hydraulic push-off unit assembly TU180	8	0.00	60.00	0.00	10.62	10.62
Highlift cat 3 mount	8	0.00	60.00	0.00	3.54	3.54
Highlift cat2 mount	8	15.00	0.00	3.54	0.00	0.89
Coupling systems	8	0.00	45.00	0.00	5.31	3.98
Valve mount	8	25.00	25.00	1.77	5.31	2.95
Sideshift	8	30.00	30.00	1.77	1.77	1.77
Toplink	8	10.00	10.00	1.77	1.77	0.59
Electrical constrols for options	8	10.00	10.00	7.08	7.08	2.36
Extention cord	8	0.00	5.00	0.00	5.31	0.44
MPU mount	2	20.00	0.00	1.77	0.00	0.59
HPU mount	2	45.00	20.00	5.31	10.62	7.52

- Explanation task "hoses for options" (estimation)

The total average workload (per month) of task 'hoses for options' consists of two parts: workload and extra workload. The workload is calculated by task time * frequency of the task (per month) (A1). The extra workload is the extra time needed to attach hoses when multiple hoses for options are installed on one machine, since the time does not go linearly up when a product has multiple options. The extra workload is estimated with the formula: task time per option (B1) * average number of configured products (per month) (B2) * average number of options per product (B3). Variables A1, B1, B2, B3 consider a multiplication factor 'x'.

Explanation task "hoses for options"	
A1 = Workload task = task time * frequency task (per month	5.17
B1 = Time per option (hours)	0.30
B2 = Average number of configured product (per month)	21.38
B3 = Average number of options per configured product	6.64
Estimated extra time = B1 * B2 * B3 (hours per month)	41.87
total process time Task "hoses for options" (hours)	47.03

A.2. Calculation configured TUs and TU180-XL

A list of the products from 2022, where for each unique product it was known how many options a product had and which options were on the product. This research also defined the process times and tasks. We can link tasks to individual products, and thus calculate the total average process time of a product. In this appendix, we provide a part of the table with the process times of a product to illustrate how the calculations were made. The columns 'process time (min)', 'Testing + hose time (min)' and 'Total time per TU (hours)' consider a multiplication factor 'x'. We explain the columns:

1. Unique product code: represents a unique product in the year 2022
2. Machine: indicates the type of machine
3. Number of options: indicates the number of options on the machine
4. Process time of the machine: summing up the task times of all tasks (it also includes different options at a product)
5. Testing and hose time: (testing + hose time) * number of options (assuming linearity of the time needed for x number of options)
6. Total time = process time + testing + hose time

Unique product code	Machine	Number of options	Process time (min)	Testing + hose time (min)	Total time per TU (hours)
SM0003306	TU180	5	867.3	177	17.405
SM0003319	TU180	4	867.3	177	17.405
SM0005622	TU	3	1230.15	70.8	21.6825
SM0005834	TU180	5	1123.95	212.4	22.2725
SM0005835	TU	2	1123.95	212.4	22.2725
SM0006014	TU	5	1495.65	247.8	29.0575
SM0006015	TU180	5	1495.65	247.8	29.0575
SM0006065	TU	3	1230.15	70.8	21.6825
SM0006125	TU180	5	1070.85	177	20.7975
SM0006133	TU180	5	1991.25	70.8	34.3675
SM0006140	TU180	5	761.1	106.2	14.455
SM0006159	TU	2	725.7	70.8	13.275
SM0006202	TU	2	991.2	141.6	18.88
SM0006203	TU180	4	991.2	141.6	18.88
SM0006237	TU180	2	672.6	70.8	12.39
SM0006246	TU	7	2309.85	283.2	43.2175
SM0006379	TU180	2	1991.25	70.8	34.3675
SM0006391	TU	3	885	70.8	15.93

Appendix B - Initial solutions (Excel code)

- Code to generate a list of available tasks

```
Sub Priority3_23_feb()
'counters
Dim i, j, x, y, count As Integer
Dim mogelijke_taken_beschrijving() As String
ReDim mogelijke_taken_beschrijving(1 To 30) As String
count = 1

For i = 1 To 32 'er zijn 32
   'If Application.WorksheetFunction.count(Worksheets("Priority3_23feb").Range("A2:A30")) <> 30 Then
        If Cells(i + 1, 5).Value = False And Cells(i + 1, 6).Value = True Then 'taak niet gedaan maar wel mogelijk\
            mogelijke_taken_beschrijving(count) = Worksheets("Priority3_23feb").Cells(i + 1, 3)
            count = count + 1
        End If
    'Else
     '   Exit Sub
    'end If
Next i

i = 0
'als er geen mogelijke taken meer zijn sluit de sub

If WorksheetFunction.CountA(Range("A2:A32")) = 30 Then
    Exit Sub
Else
    ReDim Preserve mogelijke_taken_beschrijving(1 To count - 1)  ' Resize arrays to exclude empty values
End If
```

- Code for generating a random task

```
count = count - 1

Dim randomlyChosentaak As String

' Check if the array is not empty before generating a random index
If count > 0 Then
    ' Generates a random index between 1 and count (aantal mogelijke taken)
    randomIndex = Int((count * Rnd) + 1)

    If randomIndex > count Then
        randomIndex = count ' Set randomIndex to the last index if it exceeds the array length
    End If

    ' je kiest hier een random taak
    randomlyChosentaak = mogelijke_taken_beschrijving(randomIndex) ' Now 'randomlyChosenValue' contains the value at the randomly chosen index

ElseIf count = 0 Then 'als er een taak over is
    'Exit Sub 'ga de sub uit als er geen mogelijke taken meer zijn > er gaat iets mis hier.., hij verlaat de sub altijd of nooit
End If
```

Appendix C – Neighboring solutions

C.1. Neighbor solutions: workload distribution (results)

Work-stations	Work-load dis-tribution neighbor solution 1	Work-load dis-tribution neighbor solution 2	Work-load dis-tribution neighbor solution 3	Work-load dis-tribution neighbor solution 4
1	19.67	35.00	34.00	35.66
2	76.00	76.00	76.00	76.00
3	32.16	35.25	32.25	36.00
4	27.00	34.92	32.16	34.08
5	35.67	24.17	30.17	27.33
6	30.67	31.08	28.50	28.83
7	36.58	33.92	28.58	35.92
8	35.07	22.49	31.15	18.99

C.2. Task distribution of current situation and proposed solution (Sol3)

	Task distribu-tion of cur-rent situation	Task distribu-tion of pro-posed solution
Pre-assembly components	0	0
Walking chart	1	1
Coupling plates	1	1
Assembly main frame	2	2
Prepare for ES (and valve)	2	3
Assembly knife system + Mount knife system to main frame	3	4
Assembly Turbofeeder and testing Turbofeeder	4	3
Mount Turbofeeder to main frame	6	8
Pre-assembly control block	5	1
Control block to TU	5	3
Hoses	5	5
Hoses for options	5	8
Hoses for options	8	8
Testing TU	6	6
Protection plates on knife system	6	6
Testing options	6	8
Testing options	8	8
Protection plates on the product (and cleaning)	7	7
Assembly electrical system (ES)	8	1
Mounting ES to the product	8	3
Highlift cat3 assembly and mount	8	8
Highlift cat2 assembly and mount	8	8

Valve assembly	8	1
HPU separately sold	8	1
MPU separately sold	8	1
Highlift separately sold TU180-XL	8	1
Hydraulic push-off unit assembly TU180-XL	8	5
Coupling systems	8	8
Valve mount	8	3
Sideshift	8	8
Toplink	8	8
Electrical controls for options	8	1
Extension cord	8	8
MPU mount	2	3
HPU mount (TU/TU180-XL)	2	7

Appendix D - Throughput time improvement calculations

D.1. Calculation for the relation between $E(W)$ and ρ

Rho values	U values	V = 1	T (lead time)	E(W) (T=81.25)
0.10	0.11	1.00	81.25	9.03
0.15	0.18	1.00	81.25	14.34
0.20	0.25	1.00	81.25	20.31
0.25	0.33	1.00	81.25	27.08
0.30	0.43	1.00	81.25	34.82
0.35	0.54	1.00	81.25	43.75
0.40	0.67	1.00	81.25	54.17
0.45	0.82	1.00	81.25	66.48
0.50	1.00	1.00	81.25	81.25
0.55	1.22	1.00	81.25	99.31
0.60	1.50	1.00	81.25	121.88
0.65	1.86	1.00	81.25	150.89
0.70	2.33	1.00	81.25	189.58
0.75	3.00	1.00	81.25	243.75
0.80	4.00	1.00	81.25	325.00
0.85	5.67	1.00	81.25	460.42
0.90	9.00	1.00	81.25	731.25
0.95	19.00	1.00	81.25	1543.75

D.2. Kingman example calculation

Task "HPU" workload	4.25

Workstation 2 start situation (before moving task)	
Workload	81.25
Lead time (=workload)	81.25
Utilization (rho) = workload/ available capacity	0.81
Available capacity	100.00
U ratio = rho / (1-rho)	4.33
E(W) = V * U * T = 1 * U * T	352.08

Workstation 2 end situation (after moving task)	
Workload	77.00
Lead time (=workload)	77.00
Utilization (rho) = workload/ available capacity	0.77
Available capacity	100.00
U ratio = rho / (1-rho)	3.35
E(W) = V * U * T = 1 * U * T	257.78

Workstation 7 start situation	
Workload	34.00
Lead time (=workload)	34.00
Utilization (rho) = workload/ available capacity	0.34
Available capacity	100.00
U ratio = rho / (1-rho)	0.52
E(W) = V * U * T = 1 * U * T	17.52

Workstation 7 end situation	
Workload	38.250
Lead time (=workload)	38.250
Utilization (rho) = workload/ available capacity	0.383
Available capacity	100.000
U ratio = rho / (1-rho)	0.619
E(W) = V * U * T = 1 * U * T	23.693

Graph E(W)	Start situation E(W) (before moving task)	End situation E(W) (after moving task)	Difference E(W)
Workstation 2	352.08	257.78	-94.30
Workstation 7	17.52	23.69	6.18
Net change (workstation 2 and 7 combined)	369.60	281.48	-88.12

D.3. Sensitivity analysis calculation

Utilization of workstations of the current situation (rho)

Cases	Actual workload of current situation	Available capacity W2	Utilization W2 (actual workload W2 / available capacity W2)	1	2	3	4	5	6	7	8
-	-	-	-	-	-	-	-	-	-	-	-
Case1	81.25	90.00	0.90	0.23	0.90	0.36	0.29	0.39	0.34	0.27	0.45
Case2	81.25	92.50	0.88	0.22	0.88	0.35	0.28	0.38	0.33	0.26	0.44
Case3	81.25	95.00	0.86	0.22	0.86	0.34	0.27	0.37	0.32	0.26	0.43
Case4	81.25	97.50	0.83	0.21	0.83	0.33	0.27	0.36	0.31	0.25	0.42
Case5	81.25	100.00	0.81	0.21	0.81	0.32	0.26	0.35	0.30	0.24	0.41

0.47

Utilization of workstations of solution 3 (rho)

Cases	Actual workload of solution 3	Available capacity W2	Utilization W2 (actual workload W2 / available capacity W2)	1	2	3	4	5	6	7	8
-	-	-	-	-	-	-	-	-	-	-	-
Case1	76.00	90.00	0.84	0.38	0.84	0.36	0.36	0.34	0.32	0.32	0.35
Case2	76.00	92.50	0.82	0.37	0.82	0.35	0.35	0.33	0.31	0.31	0.34
Case3	76.00	95.00	0.80	0.36	0.80	0.34	0.34	0.32	0.30	0.30	0.33
Case4	76.00	97.50	0.78	0.35	0.78	0.33	0.33	0.31	0.29	0.29	0.32
Case5	76.00	100.00	0.76	0.34	0.76	0.32	0.32	0.30	0.29	0.29	0.31

Cases	Utilization W2 of current situation	Utilization W2 of solution 3	Procentual change of utilization W2	Through put time of current situation	Through put time of solution 3	Procentual change of through put time	Graph
Case 1, ρ = 0.9	0.90	0.84	-6.46%	1159.65	820.92	-29.21%	29.21%
Case 2, ρ = 0.88	0.88	0.82	-6.46%	987.10	753.70	-23.65%	23.65%
Case 3, ρ = 0.86	0.86	0.80	-6.46%	875.92	703.30	-19.71%	19.71%
Case 4, ρ = 0.83	0.83	0.78	-6.46%	797.92	663.94	-16.79%	16.79%
Case 5, ρ = 0.81	0.81	0.76	-6.46%	739.93	632.23	-14.56%	14.56%

Estimated cycle and througput times of the cases (starting situation) (cycle time = workload + EW) (EW = U * T) (U = rho / (1- rho))

	Utilization W2	1	2	3	4	5	6	7	8	Throughput time estimate (hours per month)
workload distributie	-	20.5	81.25	32.17	26.00	35.00	30.33	24.33	40.73	
Case1	0.90	26.55	835.71	50.07	36.56	57.27	45.75	33.34	74.40	1159.65
Case2	0.88	26.34	668.06	49.32	36.17	56.30	45.13	33.01	72.77	987.10
Case3	0.86	26.14	561.36	48.64	35.80	55.42	44.55	32.71	71.30	875.92
Case4	0.83	25.96	487.50	48.01	35.45	54.60	44.03	32.42	69.95	797.92
Case5	0.81	25.79	433.33	47.43	35.14	53.85	43.53	32.15	68.72	739.93

Estimated cycle and througput times of the cases solution 3 (cycle time = workload + EW) (EW = U * T) (U = rho / (1- rho))

	Utilization W2	1	2	3	4	5	6	7	8	Throughput time estimate (hours per month)
workload distributie	-	33.99	76.00	32.25	32.16	30.16	28.50	28.58	31.15	
Case1	0.84	54.62	488.57	50.26	50.04	45.36	41.71	41.88	47.64	820.92
Case2	0.82	53.74	426.06	49.51	49.30	44.75	41.19	41.36	46.97	753.70
Case3	0.80	52.93	380.00	48.82	48.62	44.19	40.71	40.88	46.35	703.30
Case4	0.78	52.18	344.65	48.19	47.99	43.67	40.27	40.43	45.77	663.94
Case5	0.76	51.49	316.67	47.60	47.41	43.18	39.86	40.02	45.24	632.23