POLITECNICO DI TORINO

Master's degree in Mechatronic Engineering

Master Degree Thesis

Analysis, Optimization and Scheduling of Bottling Production Line

Supervised by:

Prof. Stefano Alberto Malan

Prof. Dario Antonelli

Dr. Maurizio Fantino

Candidate:

Yousef AlMasri

ACADEMIC YEAR 2019/2020

ACKNOWLEDGEMENTS

We would like to express our great appreciation to Prof. Stefano Alberto Malan and Prof. Maurizio Fantino, for their valuable and constructive suggestions, their continuous support and guidance, for their patience, motivation and immense knowledge during the planning and development of this research work. Their willingness to give their time so generously has been very much appreciated.

Also, We would like to thank the LINKS foundation for giving us this thesis oppurtunity. In addition, We would like to thank Prof. Dario Antonelli and the Engineer Alessandro Bortoletto for their help throughout this thesis.

Furthermore, we would also like to acknowledge with much appreciation the support we received from our families and friends who helped keep us motivated and supported throughout all our study career and life in general.

Finally, we would like to thank the Politecnico Di Torino for the particular experience and knowledge they gave us.

ABSTRACT

The purpose of this thesis is to study, analyze and improve the production process of a mass production line taking a standard beverage production line as a model to improve. The thesis included studies divided into two parts, the first part is related to studying the process of arranging the bulk production and scheduling it in the most time saving way to improve the overall output of the production line.

The second part is dedicated to the study of the bottling production line, then performing some changes with the software model to check for any chances of improvement, the suggested improvements of the model studied can be applied to different line models with some changes in the details.

SUMMARY

One of the most important resources is time, that if it is not the most important. Time cannot be stopped neither controlled, so it is important to use it as efficiently as possible.

Time in every production process is worth money, so in business language, lost time means lost money. Here immerges the need to create an efficient production process that can minimize as much as possible the time taken to produce a certain product.

Time management can be implied in many fields to improve the production process, from line development to machine improvements to scheduling.

Therefore, this thesis will be discussing the ways to develop the production process by means of scheduling and optimization.

Chapter 1 will shed the light on the production line that will be studied, and will explain the production process with the steps and the important aspects to be covered in this study.

Chapter 2 will explain briefly about scheduling models and some theories behind them so that the reader can understand the way the model is selected and how the constraints are written. After the model and the constraints are written, a solver will be used to handle the model which will allow the successful scheduling of the weekly production of a production plant.

Chapter 3 will then shift the attention to another part, which is the possible optimizations related to the physical production line itself, and the physical changes that can be done by testing many different layouts and running simulations to reach the best possible results.

The conclusion will put the pieces of the thesis together once again in a brief description to allow the reader to have an overview of the work done and better understand the way the thesis was created as one part instead of separated parts.

Table of Contents

LIST OF FIGURES

LIST OF TABLE

ACRONYMS

BFR	Bacardi Flavored RUM
RTS	Ready to Serve
AGV	Automated Guided Vehicle
PLC	Programmable Logic Controller
SPT	Shortest Processing Time
ERT	Earliest Release Time
LP	Linear Programming
MIP	Mixed Integer Programs
B&B	Branch and Bound
MTBF	Mean Time Between Failures
MTTR	Mean Time to Repair
GUI	Graphical User Interface
P	Processing Time
D	Deadline
ARI	Arrival Rate of Items
NOIPAC	Number of Output Items Per Arrival Cycle
INI	Initial Number of Items
NII	Number of Input Items
NOI	Number of Output Items

CHAPTER 1: INTRODUCTION

1.1 Introduction

This chapter will be the opening that sheds light on the production plant model that this thesis will handle, plus the details about the theories and science that will be used to develop this thesis.

1.2 Beverages Production Plant

In the following part, a description of a standard production process will take place, detailing the processes included in a sequential manner. The process usually is the same in any production plant working in the same sector, with some minor changes, but the order must be conserved. Example: you cannot cap a bottle before the filling takes place.

So the following is an assumption of a general line that operates and produces beverages with different varieties, but the study of the thesis can be applied to any food or beverage in a mass flow production line.

1.3 The Production Process

Any plant usually produces various products classified into different categories, examples to these categories in the alcoholic class of beverages are:

- Vermouth
- Liquor
- Sparkling wine
- Rum and other spirits

Each family has a quite different production process. In spite of this, many common aspects can be outlined. Each of these products undergo generally the same path that can be divided into two sub-processes: the bulk production process and the bottling production process.

The production process starts with the bulk production, which is mixing the raw materials and performing specific operations on the liquids to transform them into lots of the needed products such as wine or liquor.

After the bulk production starts the second part of the production process that is the bottling process. The bottling process is generally filling the bottles with the liquids produced and packing them into pallets. The two processes are really different and distinguished in the way of production, so further on explanation with details will be presented. In addition, the process can slightly change with the change of each product, since each product has its own requirements for the production.

This study will aim to improve the production process of all products in general, but because of the different requirements and for the sake of the scope of this thesis a certain type of product will be taken and the studies on its production process will take place.

The Rum production process will be taken as the model to be studied and developed, since it contains all the production operations, and has the needed complexity to be taken as a standard production process to be compared and matched to other products.

This process usually happens in two different buildings, one for each process, the two buildings are connected to move the finished lots of Rum from the bulk production building into the bottling process building.

1.3.1 The Bulk Production Process:

The Bulk production process is mainly transforming raw material stored in tanks to finished lots of a certain product ready to be filled bottles. This process has some varieties inside the Rum category itself, the M&R Rum product list is used as an example to show the different categories rum beverage can have, the list is presented in Table 1.

RUM	***Rum Carta Blanca 37.5°***
RUM	***Rum Carta Blanca 40°***
RUM	***Rum Carta Blanca 43°***
RUM	***Rum Carta Oro 37.5°***
RUM	***Rum Carta Oro 40°***
RUM	***Rum Carta Oro 43°***
RUM	***Rum Carta Negra 40°***
RUM	***Rum Carta Negra 43°***
RUM	***Oakheart 45°***
RUM	***Fuego 40° 40°***
BFR	***Limon 32 %***
BFR	***Pineapple 32 %***
BFR	***Mango Fusion 32 %***
BFR	***Razz 32 %***
RTS	***Mojito 14,9 %***
RTS	***Mojito 18 %***
RTS	***Daiquiri 20 %***
RTS	***Pinacolada 14,9 %***

Table 1: Bacardi Rum and other spirits

Each of the above products has some different details in the production process since they have different flavors or different fermentation time, but generally they all follow the same steps as any beverage line.

The raw materials are transported into the bulk production building and stored in big storage tanks. Then the production process starts by selecting a tank with enough capacity to host the materials.

Before starting any mixing, the tanks must be washed. This washing might be heavy or light, depending on the consecutive batches colorings or flavors.
For BFR and RTS products, it is necessary to have intense washing, due to the presence of aromas. While for the RUM based products, the washes are less frequent and less intensive.

Once the washing is complete, the actual production process can start by transferring the materials to be mixed into the selected production tank. The fluids are mixed for a certain amount of time, then the mixture is left to saturate and ferment for a period of time ranging between 24 to 48 hours depending on the product type.

After that, some control is done to check if the percentage of alcohol in the product is matching the standards. If no correction is needed, the mixture is filtered and moved to the pre-bottling tank.

As mentioned before, the buildings are connected and mainly plants connect buildings throughout pipes to transfer the material between buildings. These pipes also need rinsing with water before transporting the lots produced to eliminate any residues from the previous transfers.

The described process is, as any other production process, subject to downtimes that are the proportions of a time-span during which a system is unavailable. In the bulk production process, downtimes are not critical, since they can be handled without significant delays in the production total completion time. In fact, the most frequent downtimes during this part of the production process are due to small failures in components such as valves or filters that can be replaced in a relatively short amount of time without any crucial delays. Backup tanks are always available in case of any failure that cannot be solved during idle times (when the mixture is being fermented and matured).

Each lot produced in the process must be bottled and packed as one unit, which is given a unique code. This code will allow the tracking of the products and their expiry date, and will work as a safety mechanism allowing the plant to identify and separate any corrupted or poisoned batch. This constraint means that the production of the lots may have a different size than the requested amount, since the tank must be emptied completely before starting any new production cycle, even if the following batch is the same product. Any excess production is stored in the warehouse for future market needs.

It is useful also to mention that usually the bulk production runs according to a certain schedule that is weekly set to decide the best order for the production.

Figure 1 is a time vs process evolvement representing a cycle of the bulk production process with some details and indicators. The process starts from filling the tank with raw materials and ends with bottling, showing the discontinuity along the path caused by the stops in the bottling production process. The tank levels do not decrease when a delay is happening.

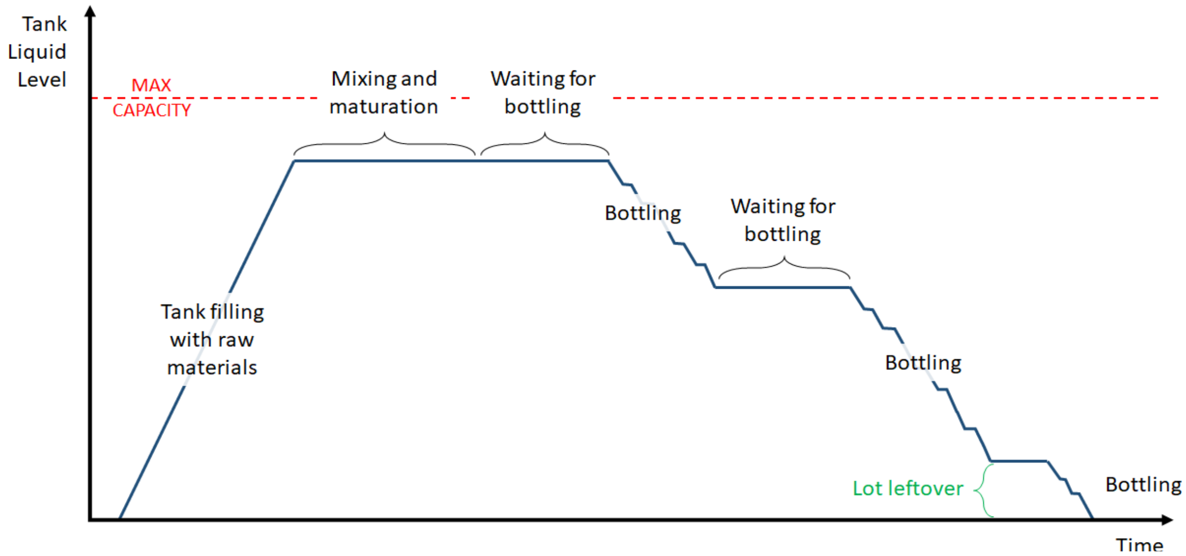

Figure 1: Tank level throughout production cycle

After the bulk production process ends, the bottling production process, which is explained with sufficient details in part 1.3.2, will take place.

1.3.2 The Bottling Production Process:

After the bulk production process ends, and the materials are transported to the bottling building to start the bottling process, a standard production line usually is divided into the following steps:

1. **De-palletization of the bottles, and setting the bottles on the conveyer**
2. **Bottle rinsing**
3. **Bottle filling**
4. **Bottle capping**
5. **Bottle drying**
6. **Bottle labeling**
7. **Bottle packing and palletization**
8. **Identification and labeling of the pallets**

After the pallets are labeled and identified they are sent to the warehouse for later shipment and market distribution.

Noting that after each one of these steps a process of quality control is performed to reject the faulty bottles and move it for treatment.

Some of these steps are machine performed, where before these machines a buffer might be found to collect the bottles and line them up for the next machine. The buffers are placed to compensate for the speed difference between machines, and to prevent the stopping of the production in case of any downtime of a machine.

The steps will be expanded to explain further the production process:

1. **De-palletization:** bottles are received as pallets, they are loaded into the de-palletizer machine that automatically places the bottles removed from the pallets onto the conveyer.
2. **Rinsing:** all the bottles must be rinsed and sanitized to remove any contamination on the inside and the outside of the bottle, then bottles are dried mostly using a jet air.
3. **Filling:** once the cleaning is complete the bottle is ready for filling. The machine is connected to the tank containing the bulk product, and the bottles are filled and monitored using a sensor that tracks the filling quantity. Any change in the bottle size requires a manual change of the machine head, this is called the Bottle Setup Delay.
4. **Capping:** Usually rinsing, filling and capping happens in the sealed mono-block to prevent the contamination from the exterior surrounding, the machine interior is isolated from the work environment around it. A quality check is performed after the capping of the bottle.
5. **Drying:** the bottles are dried to prepare them for the labeling.
6. **Labeling:** the labels are glued to different places on the bottle, each bottle having a certain label depending on the targeted market, or depending on a certain style if any special event is present. The changing of the labels is done manually changing the machine label head, this is called the Label Setup Delay.
7. **Packing and palletization:** the bottles are inserted in boxes with different dimensions depending on the bottle size, pallets are then created from the collected boxes and wrapped.
8. **Packets labeling:** wrapped pallets are labeled and sent using AGVs or other transport methods to the storage warehouse.

The transporting of the bottles between the machines is done using conveyer belts with different sizes depending on the moved items (bottles, boxes, pallets).

Some machines might face intended or un-intended downtimes, in such times the production must keep working so the conveyers usually deliver the bottles to buffers that accumulate the items and feeds the machines at its own speed. Regarding the machine speeds, the machines usually operate on predefined constant set speed, with a measure unit bottles/hour. The control of the machine speeds is automatically managed by PLC (programmable logic controller), the PLC computes the speed to operate the conveyers and the machines in order to yield the highest nominal speed of the line that is usually set by the line operator.

The speed of the line and the output produced is usually affected by the downtimes happening in the production line, which can be modeled through a stochastic process. These interruptions of the machine operations are caused by several factors, to understand better the speed of the bottling production process we need to analyze the machines downtimes, where the real speeds can be plotted in a graph. This analysis must be conducted at steady-state (i.e. without changing the type of processed product or other variables) and for a sufficient long time period in order to obtain statistically valid results. These results can be used to identify the critical machines that might be of a possible interest for the purpose of this study.

Figure 2 represents a normal operating cycle in a bottling process with speed of each machine.

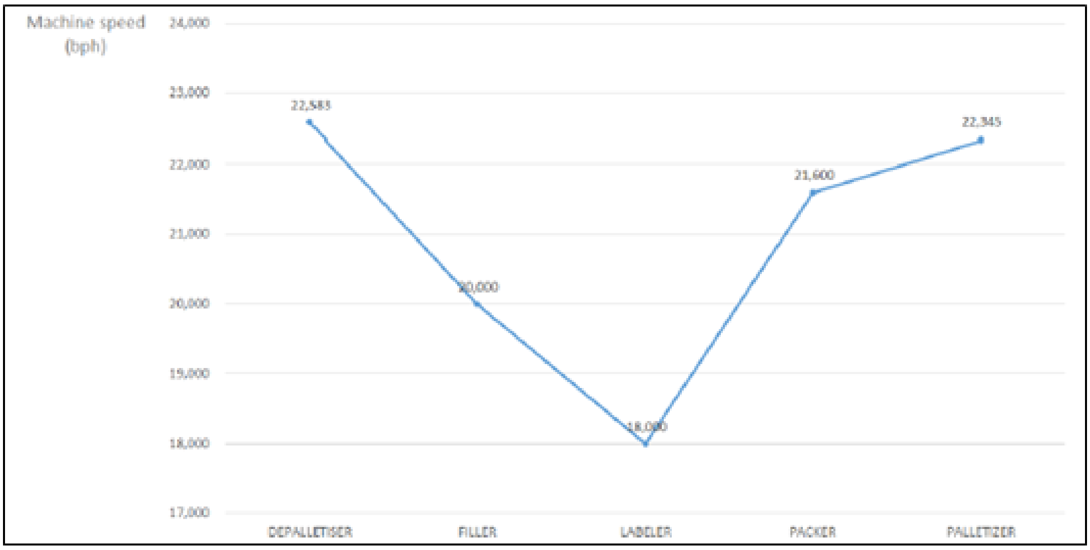

Figure 2: The machine speeds in a normal bottling cycle

1.4 The scheduling and its models

This part will go over some brief details about scheduling to highlight to the reader some important aspects that serve the scope of our study.

Mainly scheduling is effected by 3 main environments:

1. The machine environment α.
2. The job characteristics β.
3. The optimality criterion γ.

1.4.1 The Machine Environment (α):

The machine environment (α) is the type of production shop used (denoted by α_1) and the machine characteristics (denoted by α_2), yielding the following form $\boldsymbol{\alpha} = \boldsymbol{\alpha_1} \mathbf{x}\, \boldsymbol{\alpha_2}$ to define the machine environment.

Production shop $\boldsymbol{\alpha_1}$ is the type of production line installed. It can be divided into 2 main sections: where the job (j) to process with processing time (p_j) consists of a single operation that can be handled on any machine (i) or multiple operations following either a specific or nonspecific order.

While machines characteristics $\boldsymbol{\alpha_2}$ will only present the number of machines.

1.4.1.1 Single Operation Jobs:

In this part $\boldsymbol{\alpha_1}$ ϵ {∅, P, Q, R} such that:

- **∅:** One machine only in the production line → p_{1j}=p_j.
- **P:** Multiple identical machines line → p_{ij}=p_j $\forall M_i$.
- **Q:** Uniform parallel machines. Where each machine has its own speed, treating all jobs at that same speed → p_{ij}=p_j/s_i where s_i is the speed of machine i.
- **R:** Unrelated parallel machines. Where each machine has a different speed for processing the same job j → p_{ij}=p_j/s_{ij} where s_{ij} is the speed of job j on machine i.

1.4.1.2 <u>Multiple Operations Jobs:</u>

In this part $\boldsymbol{\alpha_1}$ ϵ {O, J, F} such that:

- <u>O (Open shop):</u> If each job consists of a set of operations that has to be processed on the machine but with no strict order.
- <u>J (Job shop):</u> If each job consists of a set of operations and each operation must be processed on a specific machine, and the path of jobs on machines is not identical for each job.
- <u>F (Flow shop):</u> if each job with its operation need always to follow the same path on the same set of machines.

1.4.1.3 <u>Number of Machines (α_2):</u>

When this number is positive, it represents the number of machines in the production line that is used in the scheduling problem.

While an $\alpha_2 = \emptyset$ represents a variable number of machines in the problem.
<u>N.B:</u> $\alpha_1 = \emptyset \leftrightarrow \alpha_2 = 1$.

<u>Example:</u> α= F2, where F is $\boldsymbol{\alpha_1}$ and 2 is $\boldsymbol{\alpha_2}$.

1.4.2 <u>Job Characteristics (β):</u>

The job characteristic has three options to follow; $\boldsymbol{\beta}$ ϵ {pmtn, prec, r_j}.

- **<u>Preemption (pmtn):</u>** The processing of any job can be interrupted and then resumed at a later time.

- **<u>Precedence (prec):</u>** The relation between processing of jobs exists, so a job A must be finished to allow the starting of Job B.

- **<u>Release Times (r_j):</u>** Release times of all jobs are specified and the processing cannot start before that time.

1.4.3 <u>The Optimality Criterion (γ):</u>

The Objective Function of the scheduling is the target to be reached either by minimizing or maximizing the parameter of interest, such as Maximum completion time (Cmax), Sum of completion times (ΣC), Maximum lateness (Lmax).

<u>Example:</u> job J1 (p=5, d=5), job J2 (p=10, d=15).

If the Optimality Criterion is minimizing the lateness, the jobs must be ordered as follows: J1→J2 so that all jobs are done before their deadlines, so their lateness is zero. Switching the order to J2→J1 will induce a lateness for J1.

In case the Optimality Criterion is minimizing the Completion time, then both orders will lead to the same result.

Examples of Problems:

- **1|prec|Cmax:** problem of minimizing the maximum completion time on a single machine production line, with precedence constraints.
- **Q|pmtn|ΣCj:** problem of minimizing the sum of completion times on a random number of uniform parallel machines with preemption.

1.4.4 Scheduling Models:

Scheduling Models take a set of the previous parameters, create a problem and then apply one of the theories of the scheduling problems to give a solution, some problems are linear and are polynomially solvable problems, while others are Non polynomial and require non-linear methods to solve. For the purpose of our scope this part will explain only polynomially solvable problems.

Some examples of the 'Easy' scheduling models (Pinedo, 2002):

- $\mathbf{1| | \sum C_j}$: The objective function is to minimize the summation of completion time of jobs. The problem can be solved by the SPT rule (Shortest processing Time) the rules orders the jobs with a non-decreasing order of processing times.
- $\mathbf{1| | \sum L_{max}}$**:** Jackson's rule implies that the best solution to minimize the maximum lateness can be obtained through scheduling the jobs with non-decreasing Due Date.
- $\mathbf{1| r_j|C_{max}}$**:** An optimal solution for this problem can be obtained through ERT method by ordering the jobs in non-decreasing order of Release Times.

When things become a bit more complex, the problem becomes 'NP-Hard' that with some relaxations can go back to be an 'Easy' problem (Pinedo, 2002).

1.5 Production line

A factory's main concern is to keep on working and updating its production lines until it reaches the perfect factory architecture. Optimizing a production line is directly connected to the company's profitability. The higher the efficiency of the line is, the higher the profitability of the company is.

When the factory is efficient and safe, having the optimal architecture, it will directly join its competitors in the industry.

It is all about creating a competitive advantage, and this can be done either by optimizing the production line or using new ideas or technologies.

It is inevitable that before starting with the optimization process, operational and financial feasibility analyses must take place taking also into consideration hidden costs so that the factory gets all the detailed data needed prior to any change to see if it is feasible or not.

There are many methods to optimize a production line. Mainly, plants head to modify the line architecture or to add new parts to it that mostly support automation.

1.5.1 Benefits of using automation in the optimization process:

- Using automation fills the labor gaps.
- Along with specific designed applications for scheduling weekly orders, it will lead to an optimal production plan reducing all delays and downtimes.
- Training employees on new machines, will surely lead to a higher rate of consistency in the quality of the product.
- It reduces worker's movement and increase the productivity.
- It also helps tracking and addressing directly the problems occurring in the factory, which prevents sudden problems and decrease downtimes.
- Mixed with AI power, all data will be combined and uploaded to clouds in order to use them in cloud computing and machine learning algorithms, which take past data and learn from them to create and adapt models to reality of production lines.

1.5.2 Modifications and their effects on the productivity and efficiency:

- Adding additional tanks parallel to the working ones so that the production starts directly when the previous one ends.
- Adding parallel labelers or installing labelers with already existing emergency setup, in order to reduce down time.

- Installing buffers parallel to machines taking the production instead of letting them go inside the machine, which gives time to the maintenance team to fix all issues in case of faults instead of the stopping the whole line.

- Adding parallel machines which offers many advantages:
  - Reduce setup time delay to zero, since parallel machines are set and ready to function before the original machines finish working.
  - They can also be kept for emergency cases to reduce downtime. In such cases, parallel machines installed are smaller than the original ones. Which is why the efficiency will decrease but at least it would not be null (stopped line).
  - In case activated simultaneously with the original line, it increases the speed of the production and increase the productivity.

1.5.3 Difficulties Encountered in the field:

Since technology improved and evolved, production lines changed dramatically and grew more complex, until they were no longer a linear sequence of tasks anymore.

So the following list of difficulties emerged:

- Different raw material quality.
- Complex isolated processes.
- Plant and lines architectures with parallel tracks.
- Loops and processes dependent on completed tasks.
- Difficulty in tracking the material in real time.
- Constrained adaptability.
- And many more regarding the information constraints (lack of knowledge or data).

CHAPTER 2: The Scheduling

2.1 Introduction

The production process in a brief description starts from the bulk production tanks holding the raw material. These tanks are connected to the mixing and fermentation tank. Before hosting any liquid, the mixing and fermentation tank gets washed for a duration of almost 30 min (avg. washing duration), then liquids arrive to the hosting tank and get mixed with aromas or flavors for about 15 min. Following the mixing, starts the maturation process that takes from 24-48 hours to finish, then the processed lot is transferred into the pre-bottling tank.

After studying the available production line, it can be seen that the model to be adopted needs to match a bulk production process with certain constraints preceding the transfer of the produced amount for further steps in the bottling production line.

The tank can be considered as a machine and the jobs need to pass on this machine with no other stops, meaning that the model so far can be chosen freely, but further on it is noticed that the batches do not have a release time, or a specific due date related to each job, then a suitable model to work with can be minimizing the sum of completion times.

This chapter will discuss the scheduling theories with their corresponding explanations and models, and will explain how the model and constraints matching our problem will be chosen and written. Then later on the model will be fed to the simulator which will present the results that serve the purpose of this thesis. This chapter also includes some testing with documented outputs to show the fruit of work in this chapter.

2.2 The Chosen Model

The linear model chosen is $\mathbf{1|rj|\sum C}$ such that one machine is available with N jobs, every job has a release time (the time when this job is available and ready to be processed in the system) and an objective to minimize of the sum of completion times.

In this problem all jobs are available at the beginning so rj=0 and the processing time in each case is not available but equivalent somehow to the delays and washings applied.

So the model becomes $\mathbf{1||\sum C}$ that is an easy scheduling problem, which can be solved by any sequence placing the jobs in the non-decreasing order of processing times.

The optimal solution is given by the SPT rule (**S**hortest **P**rocessing **T**ime), since it orders the jobs in the way that yields the minimum summation of completion times using the FICO Xpress solver (Xpress-Optimizer Reference Manual, 2017).

2.3 Constraints

Studying the production process stages of a production process and the duration taken by each stage were needed in order to extract the constraints and translate them into the MOSEL scheduling language (Xpress-Mosel Reference Manual, 2014).

The following are the constraints taken into consideration to give the most suitable and real model without increasing much the complexity:

2.3.1 Lots:

Due to health worries, it is obligatory to separate each lot of the production from the others to allow the tracking of the faulty lot in case any poisoning or problem happen.

Hence the first constraint says that once the tank is filled, it must be fully emptied and packed as one lot, even if it is more than the needed quantity.

Excess production will be saved in storages for further client orders.

2.3.2 Washing:

The products usually are various and branched into many groups. Inside each group there might also be different options, this means different colors or flavors for each group.

Any switch will require a washing depending on the group of the previous lot.

For our point of interest, switching between groups will imply a heavy wash (60 min = 3600sec), while switching within same group requires a light wash (30 min = 1800sec).

Note: At the beginning of the process, there must be an initial light wash of the equipment.

2.3.3 Size:

Each bottle produced has certain dimensions, and each time the bottle size has to be changed, a manual process of changing the machine heads is required in order to handle the wanted size.

This process induces a delay called Size Setup Delay (90 min = 5400sec).

2.3.4 Label:

A similar process is required each time the bottle label is changed; this delay is called Label Setup Delay (80min so 4800sec).

N.B.: Delays can stack for the same lot.

The problem is finding the best scheduling for the batches taking into consideration all the constraints in order to achieve the earliest finish of the weekly production.

2.4 The Solver

The scheduling problems has been since a very long time a subject of interest in many fields, that is why the need to solve these problems emerged and solvers were created.

The solver chosen in this project is the FICO XPRESS optimizer (Xpress-Optimizer Reference Manual, 2017), generally used as a linear programming (LP) optimization solver, which is why it is a powerful solver able to easily handle the linear programming problem faced in this project.

The solver uses a programing language called MOSEL (Xpress-Mosel Reference Manual, 2014), which is why the model of the bulk production needed to be changed to the right format in order for it to be fed into the solver.

Eq 1: $$\min \sum_{j=1}^{n} C_j$$

Eq 2: $$C_j \geq r_j + p_j \qquad \forall j = 1..n$$

Eq 3: $$C_i - p_i \geq c_j - My_{ij} \qquad \forall i = 1..n, j = 1..n$$

Eq 4: $$C_j - p_j \geq c_i - M(1 - y_{ij}) \qquad \forall i = 1..n, j = 1..n$$

Eq 5: $$C_j \in R \qquad \forall j = 1..n$$

Eq 6: $$y_{ij} \in \{0, 1\} \qquad \forall i = 1..n, j = 1..n$$

Eq 1 represents the objective function of the problem: "Minimization of summation from 1 to N jobs of the completion times".

Eq 2 First constraint: "Each job can begin only after it is available", so at time zero in our case.

Eq 3 and **Eq 4** are Disjunctive constraints: "The machine can process one job at the same time only", so either job $\boldsymbol{i}$ is processed before job $\boldsymbol{j}$ or vice versa.

Eq 5 and **Eq 6** are the constraints declaring the variable $\boldsymbol{C_j}$ as real and $\boldsymbol{y_{ij}}$ as binary, since they are the variables of the problem such as $\boldsymbol{C_j}$ (the completion time of job $\boldsymbol{j}$**)** and $\boldsymbol{y_{ij}}$ **(**the binary variable allowing big M usage) (Pinedo, 2002).

After changing the model into a suitable form for the solver, the constraints related to the first processed job and the ones defining the number of lots needed for every batch were also added to the problem to allow the solver to do its job and find the best order to process the weekly given batches.

Figure 3 shows an example taken from the written code translating the third constraint to the solver format:

```
forall(i in BATCHS, j in BATCHS | i <> j and Group(i)=Group(j)and BSize(i)= BSize(j) and BLabel(i)<>BLabel(j) ) do

    C(i) >= C(j)+ Lw*L(i) + Dsl - BIGM*y(i,j);
    C(j) >= C(i)+ Lw*L(j) + Dsl - BIGM*y(j,i);
    y(j,i) = 1-y(i,j)
    writeln("L (",i,",",j,")in loop 1 is ",L(i))
```

Figure 3: Code Example

When it comes to the MOSEL written code (Xpress-Mosel Reference Manual, 2014), it is a list of 'for loops', where each loop covers a case that the solver might face when trying to find the optimal order for the production.

For example in Figure 3, the solver will enter this loop if the following three conditions related to the previously processed lot and the lot to process now are satisfied:

- If they belong to the same group of products
- If they have the same bottle size
- If they have different labels used

In this case the completion time of the lot to process must be bigger than the previously completed lot plus the following extra delays:

- **Lw*L(i):** **L**ight **W**ash (duration 30 min), multiplied by the number of lots needed to form the requested batch.
- **DSL:** **L**abel **S**etup **D**elay (duration 90 min), which is the delay imposed by the adjustments of the machines in order to change the labels.

As for the parts of BIGM*y(i, j) or BIGM*y(j, i) and y(j, i)=1-y(i, j), they are just used to order the jobs in the correct manner (Pinedo, 2002).

A BIGM constraint can also be used for stating the relationship between variables, where M is a constant set before running the code having a value large enough to take any possible value of x. The value of M needs to be chosen with care, since If M is smaller than the upper bound of x, then it might cut off viable solutions having a higher value than M. While if M is too large, the model may become numerically difficult, and will take longer periods of time to execute (Pinedo, 2002).

As mentioned before, this loop in Figure 3 covers only one case. The rest of the delays and constraints covered in part 2.3 have their own loops, but for simplicity reasons this example is enough to give the reader an idea about the way the constraints are written in MOSEL language (Xpress-Mosel Reference Manual, 2014).

2.5 Testing, Results and Future Upgrades

The testing process took place after defining every single element necessary for the scheduling problem. It covers the majority if not all the possible cases that the operator at the factory might face.

At first, small number of batches were taken into consideration in order to understand easily the output generated. Then bigger numbers got tested to check the capacity of the solver.

Small number of batches test began with taking only 3 batches with different sizes, groups and labels in order to test that every loop written in the code works well and in the right time.

This gave the results shown in Table 2 and Table 3:

Batch Size			Group			Bottle Size			Label		
10	50	30	1	2	2	1	2	1	3	2	2
100	50	300	1	2	2	1	2	1	3	2	2
50	50	50	1	2	2	1	2	2	3	2	2
10	50	30	1	2	2	2	2	2	2	2	1
10	50	30	1	1	2	2	1	3	1	2	1

Table 2: Data Entry for Testing

Position			Completion Time			Loops
1st	2nd	3rd				
2	3	1	30	150	290	4 & 5
1	3	2	60	350	470	4 & 5
3	2	1	30	60	290	1 & 8
1	2	3	30	90	200	3 & 2
3	1	2	30	180	380	7 & 6

Table 3: Results of Testing

Explanation: in the first row of Table 3, under the **position** title: number 2 refers to the batch of 50 positioned first to process, number 3 refers to the one of 30 and the number 1 refers to the batch of 10.

Then one additional batch was added and all possible scenarios were tested, which gave us the results shown in Table 4 and Table 5 :

Batch Size				Group				Bottle Size				Label				
10	50	30	40	1	1	1	1	2	2	2	2	1	1	1	1	Same Group
10	50	30	40	1	1	1	1	2	1	2	1	1	1	1	1	
10	50	30	40	1	1	1	1	2	2	2	2	1	2	1	2	
10	50	30	40	1	1	1	1	1	2	1	2	2	1	2	1	
100	50	30	60	1	1	1	1	1	2	3	1	2	3	1	1	

Batch Size				Group				Bottle Size				Label				
10	50	30	40	1	2	1	2	2	2	2	2	1	1	1	1	Same Bottle Size
10	50	30	40	1	2	1	2	2	2	2	2	2	1	2	1	
10	50	30	40	1	2	1	2	2	2	2	2	2	1	2	2	
10	50	30	40	1	2	1	2	2	2	2	2	1	2	3	4	
100	50	30	60	1	2	3	1	2	2	2	2	1	2	4	3	
10	50	30	40	1	2	1	2	2	2	2	2	1	1	1	1	Same Label
10	50	30	40	1	2	1	2	1	2	1	2	1	1	1	1	
10	50	30	40	1	2	1	2	1	1	2	1	1	1	1	1	
10	50	30	40	1	2	1	2	1	2	3	4	1	1	1	1	
100	50	30	60	1	2	3	1	2	1	4	3	1	1	1	1	
10	50	30	40	1	2	1	2	1	1	1	2	1	1	1	2	Different
10	50	30	40	1	2	1	2	1	1	2	2	1	1	2	2	
10	50	30	40	1	2	1	2	1	2	1	3	1	2	1	3	
10	50	30	40	1	2	1	2	1	2	3	4	1	2	3	4	
100	50	30	60	1	2	3	1	2	1	4	3	1	2	4	3	

Table 4: Testing Scenarios

Position				Completion Time				
1st	2nd	3rd	4th					
1	2	3	4	30	60	90	120	Same Group
3	1	2	4	30	60	180	210	
3	1	2	4	30	60	170	200	
3	1	2	4	30	60	260	290	
3	4	1	2	30	180	320	520	
3	1	2	4	30	60	120	150	Same Bottle Size
3	1	2	4	30	60	200	230	
3	1	4	2	30	60	120	230	
1	3	2	4	30	140	280	390	
2	3	1	4	30	170	340	480	
3	1	2	4	30	60	120	150	Same Label
3	1	2	4	30	60	210	240	
4	2	1	3	30	60	120	240	
3	1	2	4	30	150	300	420	
2	3	1	4	30	180	360	510	
3	1	2	4	30	60	120	320	Different
3	4	2	1	30	90	290	350	
3	1	2	4	30	60	290	490	
1	3	2	4	30	230	460	660	
3	2	1	4	30	260	520	750	

Table 5: Results of Testing

As seen in Table 4 and Table 5 the tests were divided into 4 sections (separated horizontally by a dark grey row), each containing 5 tests, where 4 of them have batch sizes less than the lot size and only one has some batches bigger in order to test the light washing between lots constraint.

In the first section, all batches have the same group while their bottle size and labels are changing. Then in the second section the bottle size is fixed while the group and label are varying. While in the third one the label is fixed with the group and bottle size changing. And then the fourth section comes with everything varying at the same time.

These 4 sections helped testing all possible scenarios, which is obvious since all completion times in the last column, corresponding to the batch done on the last position, are different.

After that, the scheduling of 10 batches was tested and gave the results in Figure 4, Figure 5 and Figure 6:

```
Number of integer feasible solutions found is 15
Best integer solution found is  3990.000000
Best bound is  3989.999610
Numerical issues encountered:
   Dual failures    :       9 out of     14791 (ratio: 0.0006)
Uncrunching matrix
Scheduling Problem
==================
Position 1 - batch 8 - Completion time = 30
Position 2 - batch 5 - Completion time = 90
Position 3 - batch 10 - Completion time = 180
Position 4 - batch 2 - Completion time = 210
Position 5 - batch 3 - Completion time = 330
Position 6 - batch 1 - Completion time = 360
Position 7 - batch 7 - Completion time = 480
Position 8 - batch 9 - Completion time = 630
Position 9 - batch 6 - Completion time = 780
Position 10 - batch 4 - Completion time = 900
```

Figure 4: Solver Results for 10 batches

N.B.: the completion times of the batches are stacked.

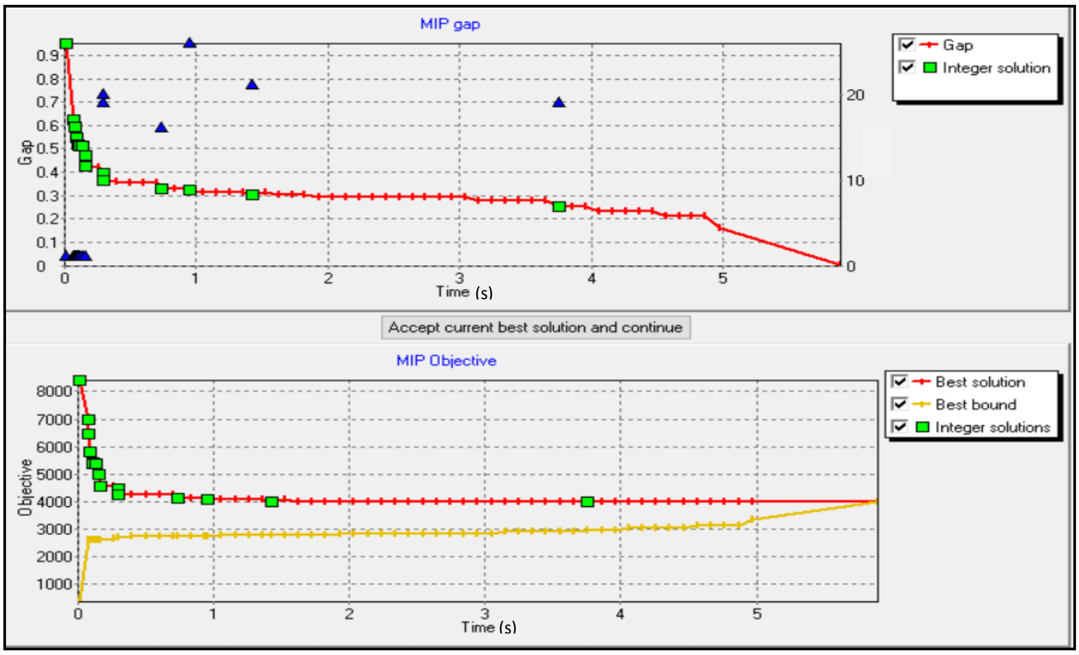

Figure 5: MIP Map for 10 batches

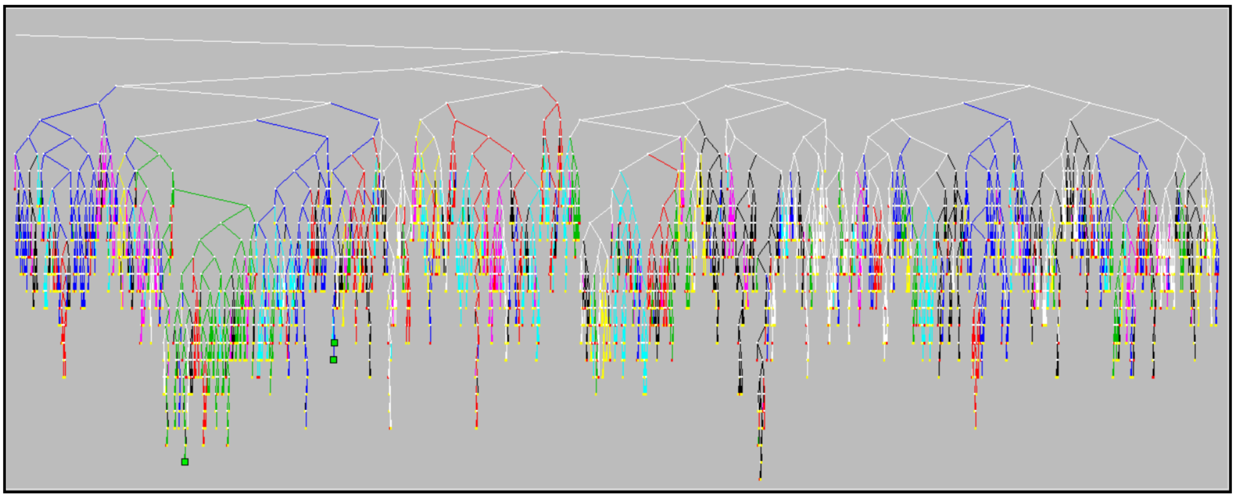

Figure 6: B&B Tree for 10 batches

As for the 15 batches, it took more time but in the end was able to present an optimal solution:

```
Number of integer feasible solutions found is 25
Best integer solution found is  8370.000000
Best bound is  6243.433841
Numerical issues encountered:
   Dual failures     :      77 out of     343291 (ratio: 0.0002)
   Singular bases    :       1 out of     217031 (ratio: 0.0000)
Scheduling Problem
==================
Position 1 - batch 4 - Completion time = 30
Position 2 - batch 2 - Completion time = 60
Position 3 - batch 8 - Completion time = 90
Position 4 - batch 13 - Completion time = 150
Position 5 - batch 5 - Completion time = 270
Position 6 - batch 3 - Completion time = 380
Position 7 - batch 11 - Completion time = 410
Position 8 - batch 7 - Completion time = 530
Position 9 - batch 9 - Completion time = 670
Position 10 - batch 1 - Completion time = 700
Position 11 - batch 14 - Completion time = 760
Position 12 - batch 6 - Completion time = 880
Position 13 - batch 15 - Completion time = 970
Position 14 - batch 12 - Completion time = 1120
Position 15 - batch 10 - Completion time = 1350
```

Figure 7: Solver Results for 15 batches

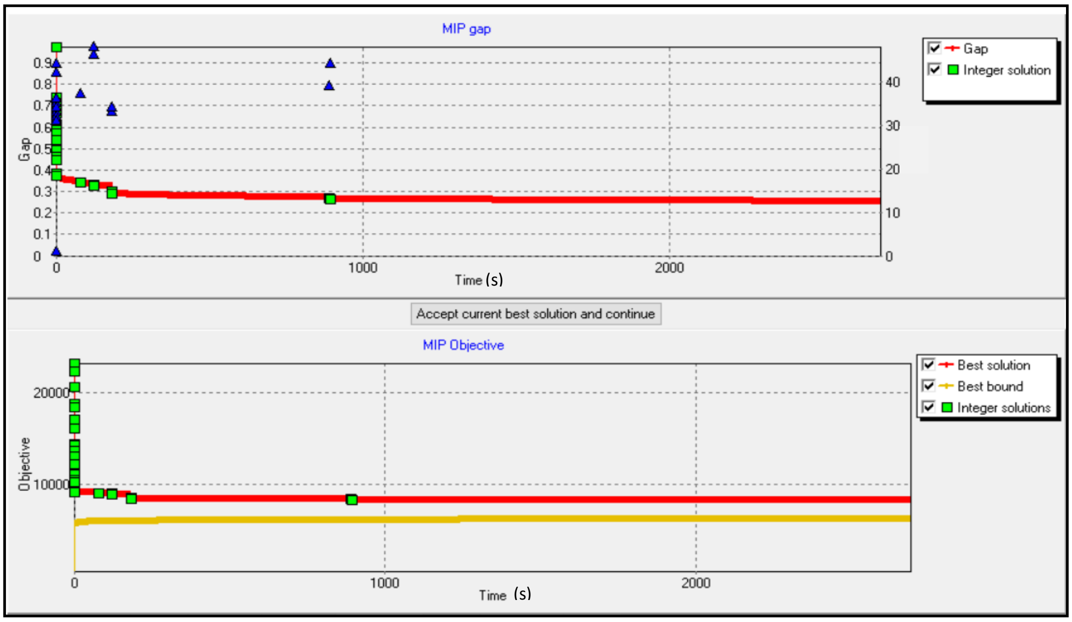

Figure 8: MIP Map for 15 batches

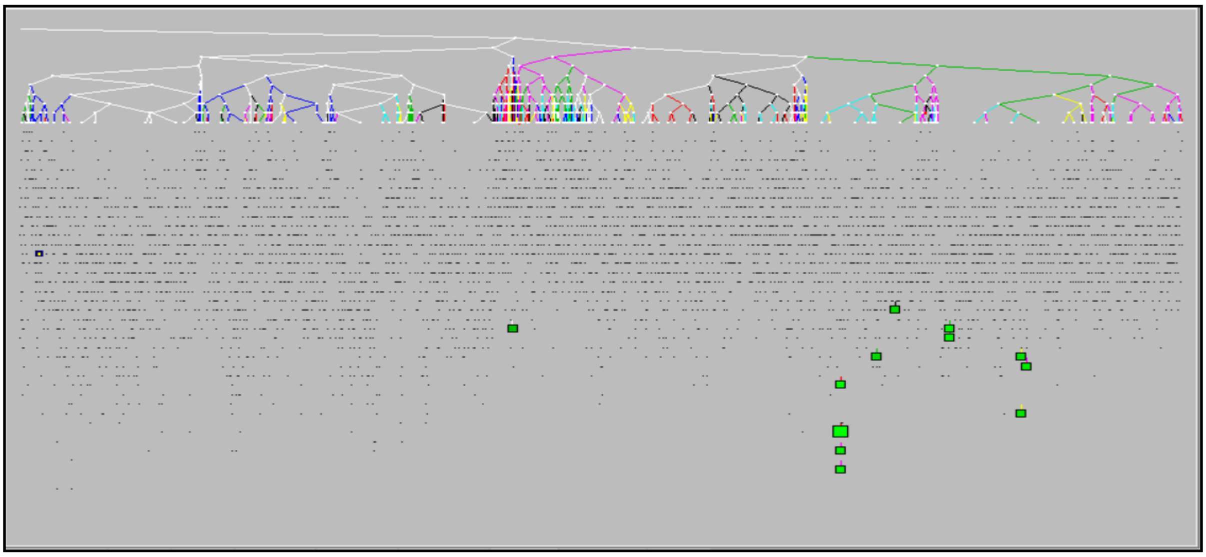

Figure 9: B&B Tree for 15 batches

As seen in the previous images (MIP and B&B), the solver used offers the output in several ways. The user can directly get data related to the order and delay corresponding to each batch, or he can go through more details like the MIP map (Mixed Integer Programs) for example that shows all integer solutions found (green squares) laying on the line of best solutions (red) while the best bound line (yellow) follows it until they meet and give the optimal solution of the problem. This occurred in the test of 10 batches, but not in the one of the 15 since the option of "accept current best solution and continue" was checked so the yellow line in this case is nothing but the worst case relaxation of all remaining nodes of the search B&B tree (Xpress-Optimizer Reference Manual, 2017).

When it comes to the "B&B Tree" so "Branch and Bound Tree", it is a process that the solver chooses when the problem becomes more complicated with a bigger number of variables, especially in the case of Big-M problems (Pinedo, 2002). It consists of going through all nodes placed on the top of the tree, checking their solution one by one (going vertically from top to bottom) in order to choose the feasible one leading to the optimal solution (lowest green square in the tree).

This whole process can become faster but a bit less effective using Heuristic methods that guide the solver to choose the nodes and get an optimal solution in less time.

Examples of Heuristics:

- <u>Branch and Bound with a time limit:</u> B&B works normally testing feasible solutions and when the time limit passes, the best solution is the one found at that moment.
- <u>Truncated B&B:</u> This one avoids going through all branches, it just takes the most promising ones according to the solver, calculates their results and chooses the best one without going back and re-discussing previous decisions.
- <u>Beam Search:</u> Is a type of Truncated B&B, but this one chooses the most promising nodes instead of branches so it narrows more the search process.
- <u>Greedy algorithms:</u> It takes a local optimal choice hoping that it will lead us to the best solution. It consists of iterations, where in each one a partial solution is generated extending the one preceding it but with a certain given rule. Example: Local Search.

2.6 Connection and Output Handling

First it is necessary to make clear that this thesis is making use of some parts from another thesis done by fellow students. The previous thesis was aiming to reach the same target, optimization of the mass production line, but was focused on modeling the production line and creating a simulator able to simulate a production process of a beverage production plant. The created software simulator runs a Monte Carlo analysis to achieve a high accuracy close to the real system, and then plots all useful results and output values. The simulator will be discussed with more details in section 3.2.

The work first started by taking the offline version of the simulator coded in JavaScript, and creating a python wrapper that handles all raw data generated as an output of this simulator. This raw data cannot be read easily and needed much effort to extract useful information, so the wrapper changed this data into an easy readable format and also provided a graphical method of presenting the results by plotting the performance of any needed element in the production line chain.

Next step was scheduling the batches before feeding them to the simulator, since the simulator runs the batches in the same order as in their text file. To do that, a certain solver called FICO Xpress (Xpress-Optimizer Reference Manual, 2017) that uses MOSEL code (Xpress-Mosel Reference Manual, 2014) is used. This scheduler takes all its inputs from a text file called "Weekly_data.txt", containing all necessary information for the production of the upcoming week and its scheduling. That is why the user must fill all these entries available in the text file then run the MOSEL code (Xpress-Mosel Reference Manual, 2014) on Fico Xpress (Xpress-Optimizer Reference Manual, 2017) to order all the jobs in the optimal way. This optimal order is extracted and printed in another text file called "mosel_outputs.txt" along with the delays and sizes corresponding to each batch.

After that, the python code "batch_feeder.py" takes all the data in "mosel_outputs.txt", translates them to the right format and prints them in the optimal order on separate text files ("BatchSizes.txt" and "BatchDelays.txt") for them to be handled by the main python code "project_1.py".

Then, "BatchSizes.txt" becomes the input of the wrapper, first part of "project_1.py", along with the production line architecture file in order to run the Monte Carlo simulation and generate some raw data that are taken by the second part of "project_1.py".

These raw data taken by the python code are translated, saved in variables then added to the delays available in "BatchDelays.txt" in order the present only the useful information and plot them.

Figure 10 will help understand the process in a better way:

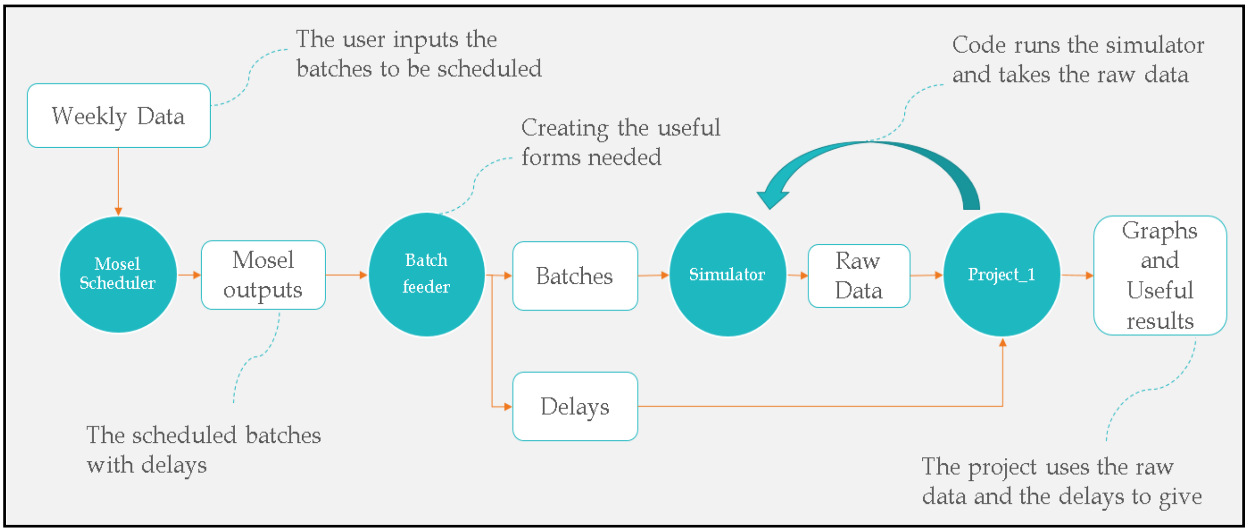

Figure 10: Scheme of the Whole Process

CHAPTER 3: Production Line Optimization

3.1 Introduction

As mentioned before in previous sections, the production line of a company is the core stone that can distinguish a successful evolving modern plant from a small side business. Once again it is necessary to emphasize that the development of production lines is a high value point of interest for plants looking to improve their business and stay within the leading companies. Some companies have specific sectors for research and development dedicated to keep ongoing studies on how to improve in their production lines.

This chapter will present the production line with the model that will be worked upon. It will also be dedicated to tests and trials, in a scientific manner, to obtain better results in the production process.

3.2 The Simulator and the Architecture

In order to develop any production line, prior researches must be done to insure that changing the production line and paying the expenses will give the required result. In order to perform these studies, models are usually designed to allow the researcher to change the parameters and components of the production line and analyze their induced changes.

In this study, the model of the production line that will be created using a simulator able to run simulate the production process with a certain line configuration having its own related properties and give the obtained results.

3.2.1 The Monte-Carlo Simulator:

The previously mentioned simulator was built as a java script, it runs taking 2 input parameters: **line model,** a JSON file that is built using the graphical user interface, and **batch sizes** which is a text file containing the size of the batches to be produced.

The simulator tries to simulate as close as possible the actual production process that of course might have unpredictable events such as setbacks, delays, machine failures. In order to simulate these unpredictable events having such a stochastic behavior, the simulator gives an option within the machine properties related to the MTBF (**M**ean **T**ime **B**etween **F**ailures) and the MTTR (**M**ean **T**ime **T**o **R**epair). These parameters are usually deduced from the data provided by the company doing the study so in this case they will be estimated. Then given the historical data of machine failures within a certain time-span, a mean can be calculated and used to improve the accuracy of the simulator.

The previous indicated parameters are not enough to give a reliable accuracy, since the delays tend to have a non-linear behavior, so multiple Monte Carlo iterations are performed to ensure that the taken results are within the wanted confidence interval.

3.2.1.1 <u>The Line Model:</u>

The line model is created on a GUI that can be accessed through any browser, it gives the user the option to add machines, buffers, connect them, alter their properties and then run the online simulator. The online simulator then runs the created model with the provided batch sizes and provides graphs containing the details about the simulated production.

Figure 11 is a screen view of the front end GUI with a built model:

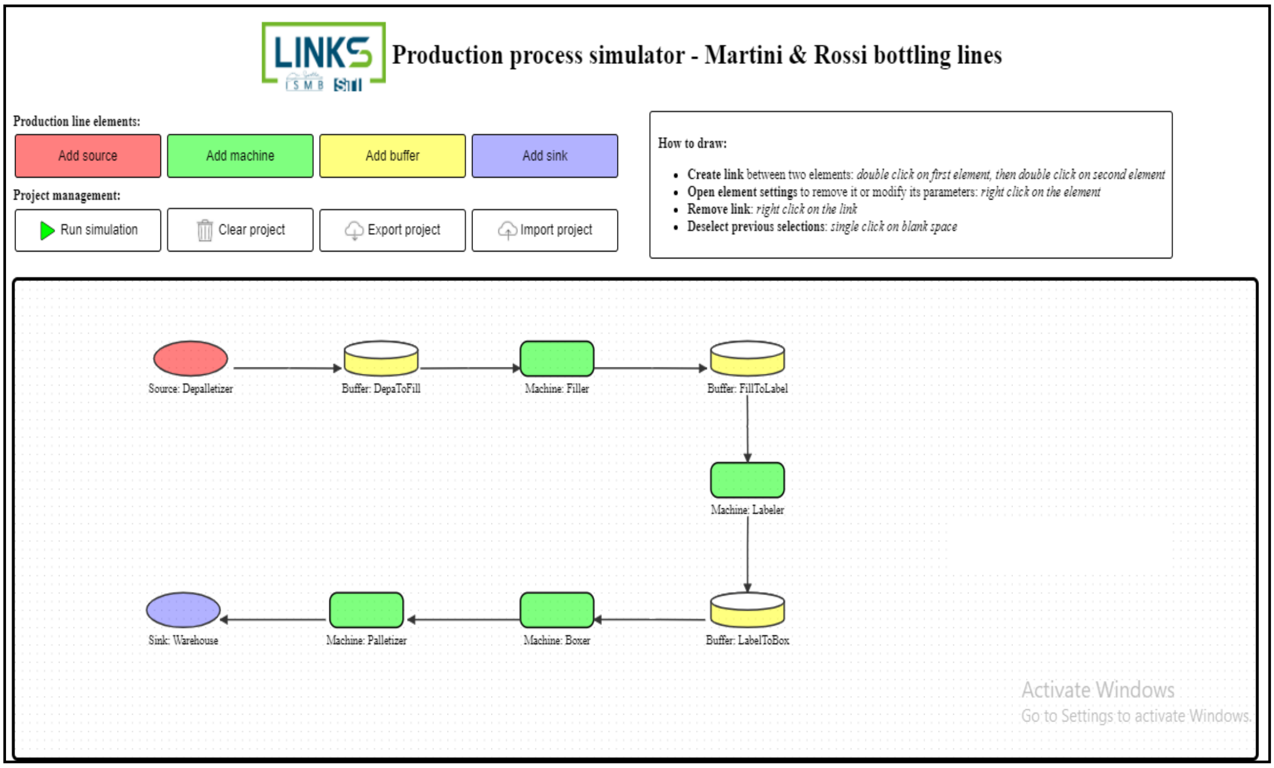

Figure 11: Front end of the simulator

Each element added to the white space is representing a certain component in the production line as shown in Figure 11 such as: de-palletizers, machines, buffers, fillers, labelers, boxers, palletizers and warehouses as endpoints.

These elements have specific parameters such as machine name, speed, processing time (speed in item/h), input and output ratios (example: each box entering consists of 10 bottles).

These parameters can be changed according to the user needs to match the model under study.

Figure 12 shows a preview of the machine properties window:

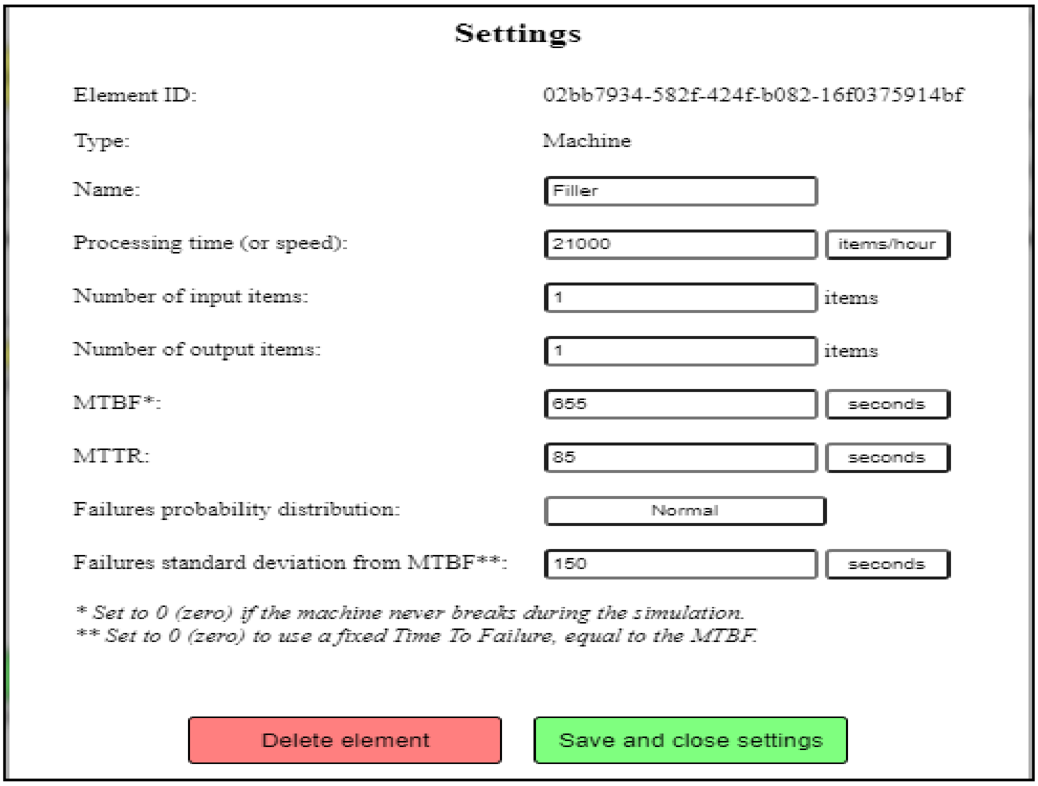

Figure 12: Machine Properties window

After setting the parameters, the user needs to run the simulation.

An offline simulator is available to perform the testing part. This offline version is a java script code that is ran through the command prompt. It creates an output text file containing raw data that cannot be comprehended easily and does not provide any graphical reading methods to facilitate the result reading and interpretation.

So as mentioned in section 2.6 before starting to work on the simulator, a method to improve readability was needed. For this reason, the Python Wrapper code was created.

Figure 13 show the raw data found the output text file created by the simulator.

```
Production lot size: 30
Simulation results: {"Containers":[{"PlaceName":"Container1","LastArrivalTime":2580.0,"Trend":[{"Marking":0.0,"Age":0.0},{"Marking":0.0,"Age":20.0},{"Marking":0.0,"Age"
3.0,"Age":660.0},{"Marking":3.0,"Age":680.0},{"Marking":3.0,"Age":700.0},{"Marking":3.0,"Age":720.0},{"Marking":3.0,"Age":740.0},{"Marking":3.0,"Age":760.0},{"Marking":
g":7.0,"Age":1380.0},{"Marking":7.0,"Age":1400.0},{"Marking":7.0,"Age":1420.0},{"Marking":7.0,"Age":1440.0},{"Marking":8.0,"Age":1460.0},{"Marking":8.0,"Age":1480.0},{"
Marking":11.0,"Age":2080.0},{"Marking":12.0,"Age":2100.0},{"Marking":12.0,"Age":2120.0},{"Marking":12.0,"Age":2140.0},{"Marking":12.0,"Age":2160.0},{"Marking":12.0,"Age
81395349,"Trend":[{"Marking":0.0,"Age":0.0},{"Marking":0.0,"Age":20.0},{"Marking":0.0,"Age":40.0},{"Marking":1.0,"Age":60.0},{"Marking":1.0,"Age":80.0},{"Marking":1.0,"
rking":3.0,"Age":720.0},{"Marking":3.0,"Age":740.0},{"Marking":3.0,"Age":760.0},{"Marking":4.0,"Age":780.0},{"Marking":4.0,"Age":800.0},{"Marking":3.0,"Age":820.0},{"Ma
},{"Marking":5.0,"Age":1440.0},{"Marking":4.0,"Age":1460.0},{"Marking":5.0,"Age":1480.0},{"Marking":5.0,"Age":1500.0},{"Marking":5.0,"Age":1520.0},{"Marking":5.0,"Age":
ng":4.0,"Age":2140.0},{"Marking":4.0,"Age":2160.0},{"Marking":4.0,"Age":2180.0},{"Marking":4.0,"Age":2200.0},{"Marking":3.0,"Age":2220.0},{"Marking":3.0,"Age":2240.0},{
1.0,"Age":140.0},{"Marking":1.0,"Age":160.0},{"Marking":0.0,"Age":180.0},{"Marking":0.0,"Age":200.0},{"Marking":1.0,"Age":220.0},{"Marking":0.0,"Age":240.0},{"Marking":
,{"Marking":1.0,"Age":880.0},{"Marking":0.0,"Age":900.0},{"Marking":0.0,"Age":920.0},{"Marking":1.0,"Age":940.0},{"Marking":1.0,"Age":960.0},{"Marking":1.0,"Age":980.0}
,"Age":1580.0},{"Marking":1.0,"Age":1600.0},{"Marking":1.0,"Age":1620.0},{"Marking":0.0,"Age":1640.0},{"Marking":1.0,"Age":1660.0},{"Marking":1.0,"Age":1680.0},{"Markin
80.0},{"Marking":1.0,"Age":2300.0},{"Marking":1.0,"Age":2320.0},{"Marking":1.0,"Age":2340.0},{"Marking":1.0,"Age":2360.0},{"Marking":1.0,"Age":2380.0},{"Marking":1.0,"A
ing":25.0,"Age":320.0},{"Marking":25.0,"Age":340.0},{"Marking":24.0,"Age":360.0},{"Marking":24.0,"Age":380.0},{"Marking":24.0,"Age":400.0},{"Marking":23.0,"Age":420.0},
0,"Age":1020.0},{"Marking":13.0,"Age":1040.0},{"Marking":13.0,"Age":1060.0},{"Marking":12.0,"Age":1080.0},{"Marking":12.0,"Age":1100.0},{"Marking":12.0,"Age":1120.0},{"
g":4.0,"Age":1720.0},{"Marking":4.0,"Age":1740.0},{"Marking":4.0,"Age":1760.0},{"Marking":4.0,"Age":1780.0},{"Marking":3.0,"Age":1800.0},{"Marking":3.0,"Age":1820.0},{"
ge":2420.0},{"Marking":0.0,"Age":2440.0},{"Marking":0.0,"Age":2460.0},{"Marking":0.0,"Age":2480.0},{"Marking":0.0,"Age":2500.0},{"Marking":0.0,"Age":2520.0},{"Marking":
.0,"Age":1380.0},{"Marking":1.0,"Age":1440.0},{"Marking":1.0,"Age":1500.0},{"Marking":1.0,"Age":1560.0},{"Marking":1.0,"Age":1620.0},{"Marking":1.0,"Age":1680.0},{"Mark
e":200.0},{"Marking":2.0,"Age":220.0},{"Marking":2.0,"Age":240.0},{"Marking":2.0,"Age":260.0},{"Marking":2.0,"Age":280.0},{"Marking":2.0,"Age":300.0},{"Marking":2.0,"Ag
ing":2.0,"Age":940.0},{"Marking":2.0,"Age":960.0},{"Marking":1.0,"Age":980.0},{"Marking":1.0,"Age":1000.0},{"Marking":2.0,"Age":1020.0},{"Marking":2.0,"Age":1040.0},{"M
e":1640.0},{"Marking":2.0,"Age":1660.0},{"Marking":2.0,"Age":1680.0},{"Marking":2.0,"Age":1700.0},{"Marking":2.0,"Age":1720.0},{"Marking":2.0,"Age":1740.0},{"Marking":2
},{"Marking":2.0,"Age":2360.0},{"Marking":2.0,"Age":2380.0},{"Marking":2.0,"Age":2400.0},{"Marking":1.0,"Age":2420.0},{"Marking":1.0,"Age":2440.0},{"Marking":2.0,"Age":
```


Figure 13: Raw Data

3.2.2 The Python Wrapper:

A python code was created for the previously listed need. This python code extracts the data from the output text file generated by the offline simulator, translates them into a readable format to the user and then plots the obtained results for further data processing.

Figure 14 shows a portion of the code written to obtain the results:

```python
dict1=json.loads(str1)                                  # creation of dictionnary
del dict1["SimulationExitCode"]
del dict1["SimulationId"]

for i in dict1.keys():
    Trend[i]=""
    for j in range(len(dict1[i])):
        list4.append(dict1[i][j]["Trend"])
        Trend[i]=list4
    list4=[]
                                                        # now we have the trends of all elements of the production line
print(f"the correct delay in min is {data1[counter]}")

sink=Trend["Containers"][0]                             # taking only the trend of containers (last element of the line)
for i in sink:
    i["Age"]=i["Age"]+data1[counter]*60                 # adding washing and setup delays in seconds


print (f"sink is  {sink}\n\n ")                         # printing results

x_axis1=[]                                              # preparing for plotting
y_axis1=[]
for i in sink:

    x_axis1.append(i['Age'])
    y_axis1.append(i['Marking'])

plt.plot(x_axis1,y_axis1,color='g',linewidth=0.5)       # plotting
plt.title('sink ')
plt.grid(True)
plt.xlabel('time')
plt.ylabel('bottles')
plt.show()

counter+=1                                              # incrementation of all variables used for iteration
jump+=3
```

Figure 14: Python Code

Figure 15 shows the results after the handling of the python wrapper:

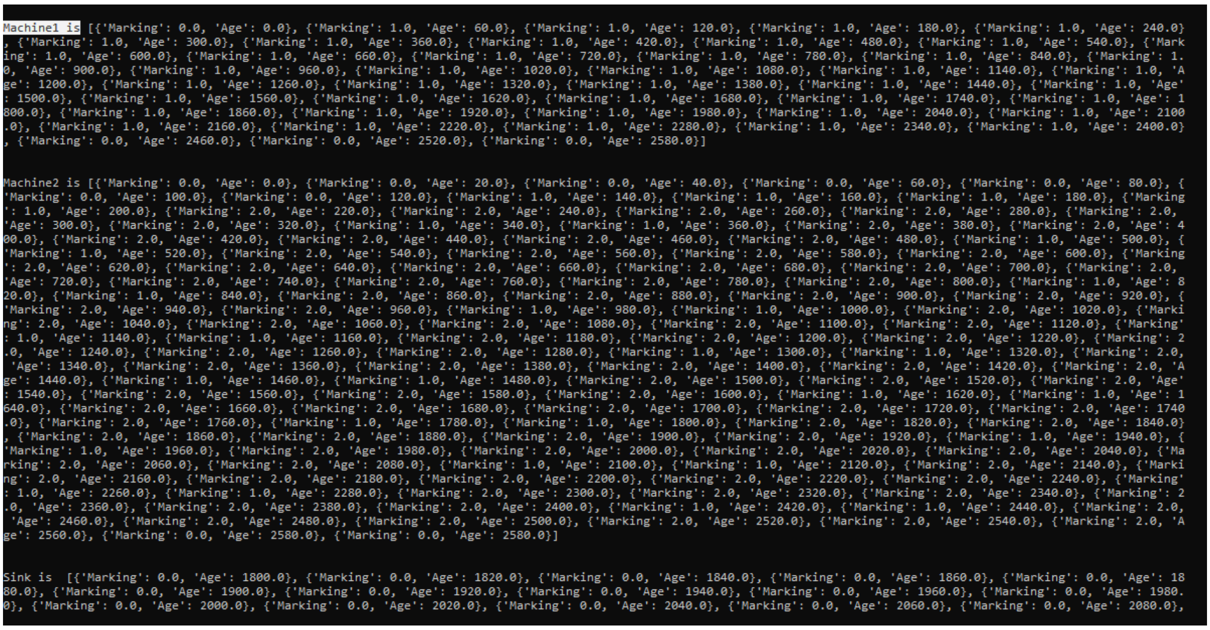

Figure 15: Treated data

As presented in Figure 15, everything is now well organized, where every batch is presented separated from the other batches instead of having a chaotic presentation of all elements simultaneously.

The results are then plotted to help the user inspect the results in a graphical way, as seen in Figure 16:

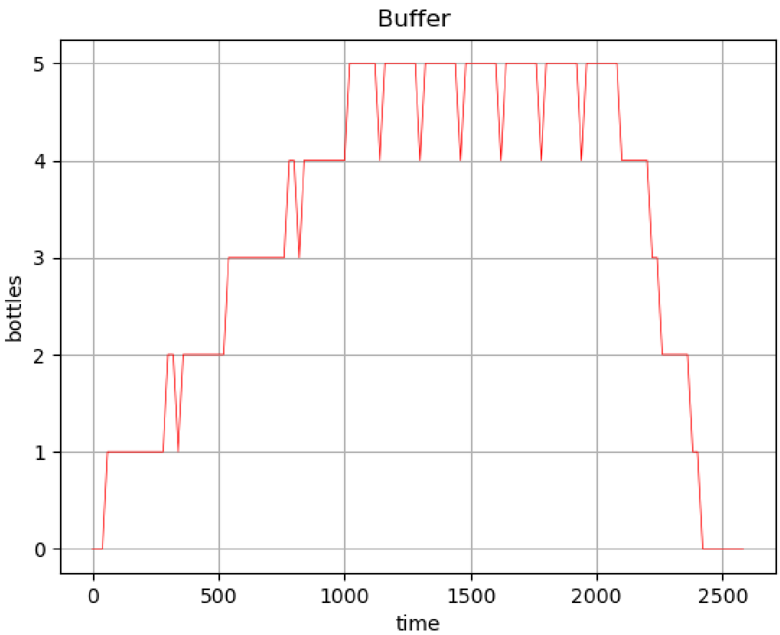

Figure 16: Buffer plot

As mentioned before in section 2.6, after reviewing the results of the simulator, the need of a missing parameter was found in order to keep a high credibility of the simulations. This parameter was the delays induced in both the bulk production process and at the beginning of the bottling process that are the washing and setup delays. These delays are simulated in the scheduling part listed before, so the python wrapper has to perform the following steps in an atomized principle:

1. Take the input files
2. Copy the output of the solver to the input files of the simulator
3. Run the simulator
4. Take the results from the output file
5. Add the delays taken from the scheduler in the correct places
6. Perform the needed changes to improve the readability of the results
7. Plot the results

After the wrapper performs this process, the results can be taken and studied. So in the next part (3.3), tests will be carried out stating the change in each iteration to seek possible improvements to the production line.

3.3 Testing and Iterations

The testing explores optimization possibilities by running several trials with different changes and analyzing the results.

The first optimization possibility is adding another machine in parallel to the bottle neck of the production line that is the Labeler, the machine with the highest MTBF between the other machines. In this way, the Labeler can have an identical version or a smaller sized one working in parallel to it, in order to speed up the production process and insure that the production line has less down times.

The second optimization possibility is by changing the buffer sizes, while checking its subsequent effect on the production process. The target of these changes is to reduce the buffer sizes, which by their turn optimize the production plant by saving valuable space.

The two above mentioned possibilities are performed by multiple runs with different batch sizes, in order to study all the possible behaviors that the production line might face.

The production line to be tested is presented in Figure 11, it is composed of 6 machines, 3 buffers and a sink that is the storage warehouse where the stock units ready for the market are stored.

The settings of the machines are constant for all tests and listed below in Table 6 and Table 7:

De- palletizer	INI	variable	ARI (Items/hour)	100	NOIPAC (Items)	100

Table 6: Source Properties

Filler	Processing time	2100	MTBF	655	MTTR	85	Standard deviation	150	NII	1	NOI	1
Labeler	Processing time	1700	MTBF	851	MTTR	109	Standard deviation	151	NII	1	NOI	1
Boxer	Processing time	3000	MTBF	454	MTTR	67	Standard deviation	17	NII	6	NOI	1
Palletizer	Processing time	3333	MTBF	617	MTTR	81	Standard deviation	19	NII	10	NOI	1

Table 7: Machines Properties

3.3.1 Identical Parallel Labeler Test:

In this part the first optimization possibility will be studied by running simulations on the production line presented in Figure 11, with a parallel labeler having the exact same properties as the original labeler.
This test is conducted with multiple batch sizes and the results are presented in Table 8 and Table 9, the unit in the tables is bottles.

The buffers capacities are: 1000, 2800 and 4500 respectively.

Batch Size	Depa-To-Fill Max Value	Fill-To-Label Max Value	Label-To-Box Max Value	Production Time
1,000	388	173	75	5min03sec
10,000	738	2135	509	46min04sec
20,000	759	2695	1355	98min59sec
30,000	761	2799	1787	146min18sec
40,000	758	2799	3013	197min02sec
60,000	755	2799	2932	274min23sec
80,000	755	2799	3750	365min05sec

Table 8: Results of Original Line

Batch Size	Depa-To-Fill Max Value	Fill-To-Label Max Value	Label-To-Box Max Value	Production Time
1,000	392	0	225	5min02sec
10,000	750	0	2341	46min04sec
20,000	729	607	4489	97min10sec
30,000	723	2550	4499	138min13sec
40,000	766	2799	4499	187min43sec
60,000	763	2800	4500	274min27sec
80,000	764	2800	4500	369min37sec

Table 9: Results with Parallel Labeler

It is easy to notice the effect of the parallel Labeler especially on the second and third Buffer, since no bottles are waiting to be processed anymore on Buffer_2 (Filler-To-Labeler) while instead they are accumulating on Buffer_3 (Labeler-To-Boxer) because the Boxer machine speed is not sufficient to handle them instantly.

3.3.2 Emergency Parallel Labeler Test:

In this part the parallel Labeler has a numerically smaller size than the original Labeler, and this is because it is used in cases of emergencies in order to prevent the stopping of the production.

The results of this test are presented in Table 10 and Table 11.

Batch Size	Depa-To-Fill Max Value	Fill-To-Label Max Value	Label-To-Box Max Value	Production Time
1,000	388	173	75	5min03sec
10,000	738	2135	509	46min04sec
20,000	759	2695	1355	98min59sec
30,000	761	2799	1787	146min18sec
40,000	758	2799	3013	197min02sec
60,000	755	2799	3432	274min23sec
80,000	755	2799	3750	365min05sec

Table 10: Results of Original Line

Batch Size	Depa-To-Fill Max Value	Fill-To-Label Max Value	Label-To-Box Max Value	Production Time
1,000	373	0	224	4min55sec
10,000	735	0	2561	46min13sec
20,000	728	530	4136	96min59sec
30,000	761	2799	4499	144min34sec
40,000	765	2800	4500	192min46sec
60,000	763	2800	4500	273min38sec
80,000	766	2800	4500	364min23sec

Table 11: Results with Emergency Labeler

The results in Table 10 and Table 11 show that a smaller size Labeler can play the exact same role and have the same effect as an identical machine with the original machine.

The effect on the production time is negligible, so such addition did not drastically improve the production process over all, but it can be a good counter measure in the case of repetitive failures in the Labeler machine.

3.3.3 50% Buffer Size Production Test:

In this part all buffer sizes are now halved in order to investigate the second optimization possibility related to decreasing the buffer sizes to save valuable space in the production plant.

Buffer Size	Depa-To-fill	Fill-To-Label	Label-To-Box
Original Size	1000	2800	4500
Reduced Size	500	1400	2250

Table 12: Buffers Old and New Sizes

	Depa-To-Fill Max Value	Fill-To-Label Max Value	Label-To-Box Max Value	Production Time	
Batch of 20,000	759	2695	1355	98min59sec	**Original**
	258	1400	1423	97min52sec	**Reduced**
Batch of 30,000	761	2799	1787	146min18sec	**Original**
	263	1400	2016	146min27sec	**Reduced**
Batch of 40,000	758	2799	3013	197min02sec	**Original**
	257	1400	1962	191min52sec	**Reduced**
Batch of 60,000	755	2799	2932	274min23sec	**Original**
	267	1400	2247	287min15sec	**Reduced**
Batch of 80,000	755	2799	3750	365min05sec	**Original**
	263	1400	2246	366min18sec	**Reduced**

Table 13: Results of 50% Reduction

Table 13 shows the effect of reducing buffer sizes on each of the buffers of the production line. N.B: red entries in Table 13 highlight full or almost full buffers during the production.

Although the simulations covered large sizes of batches, but the 50% reduction in the sizes of buffers did not negatively affect the production time by a noticable value, meaning that the buffer sizes can be decreased with a high percentages without increasing much the production processs time.

This decrease can be changed from 50% to 40% if the production plant wishes to keep a safe margin, but this simulation proves this possibilty of improvement by reducing the space in the production plant.

3.3.4 The Path to the Best Production Line Architecture:

In this part, further more studies will take place to check for optimization margins, to do that the 20,000 bottles batch is chosen as a sample. The 20K batch has an average and realistic size large enough to fulfill the needs of the study without having a very high simulation time.

- **Original Production Line:**

Presented in Figure 17, Figure 18, Figure 19, Figure 20 and Figure 21 are the results of the processing of the 20,000 bottle batch that took 98min59sec to end.

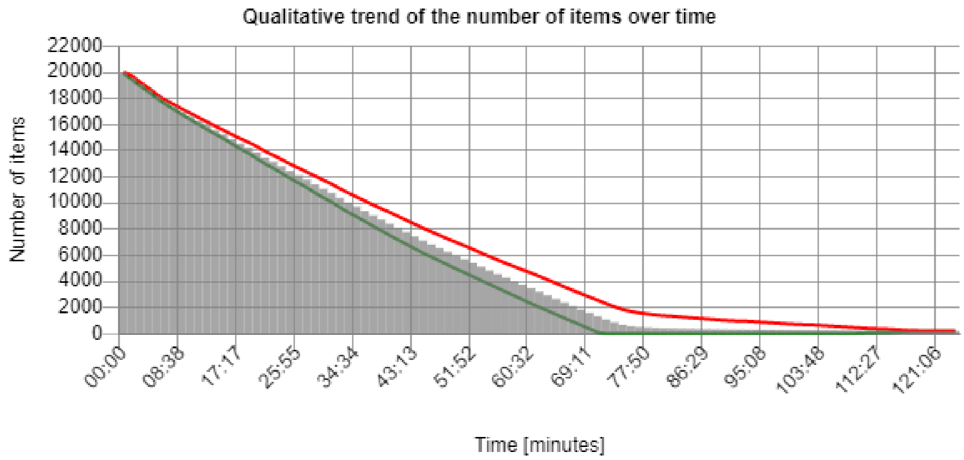

Figure 17: Graph of Depalletizer

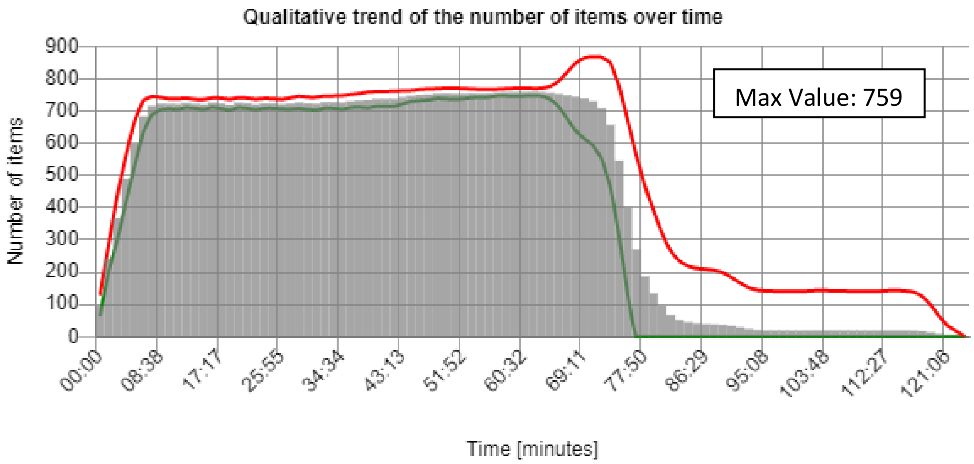

Figure 18: Graph of DepaToFill

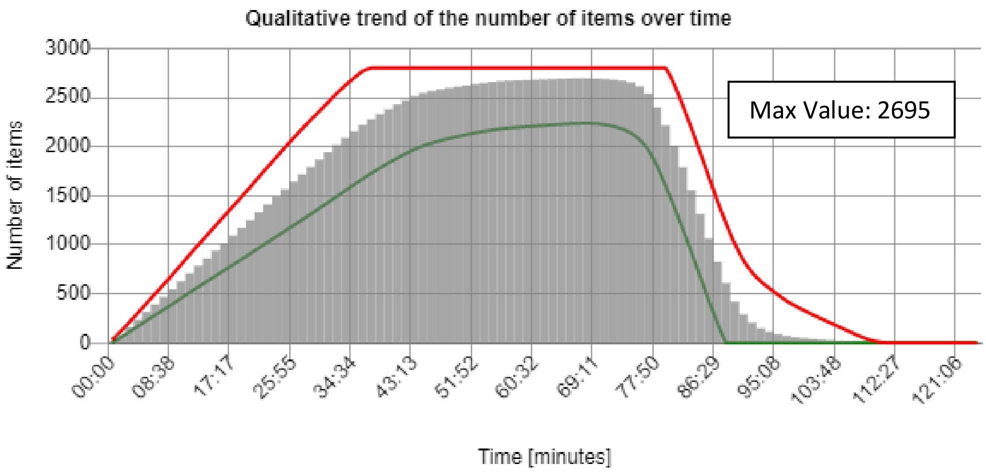

Figure 19: Graph of FillToLabel

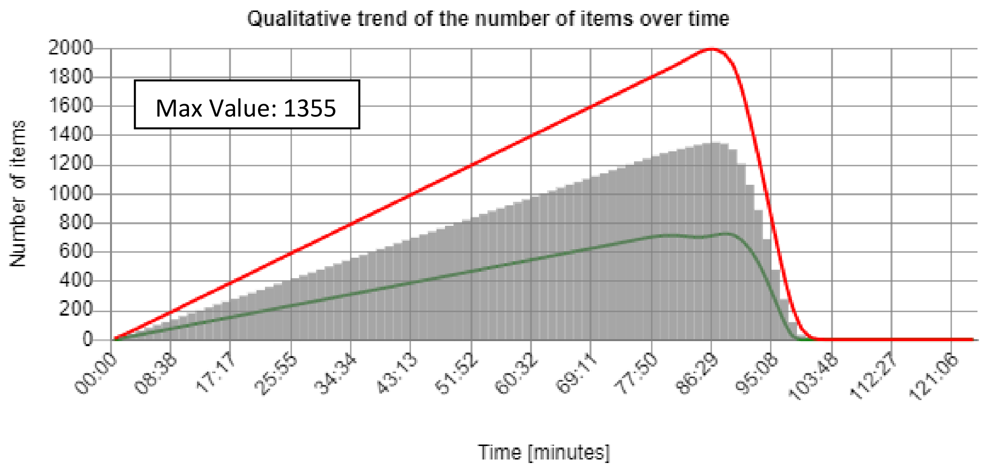

Figure 20: Graph of LabelToBox

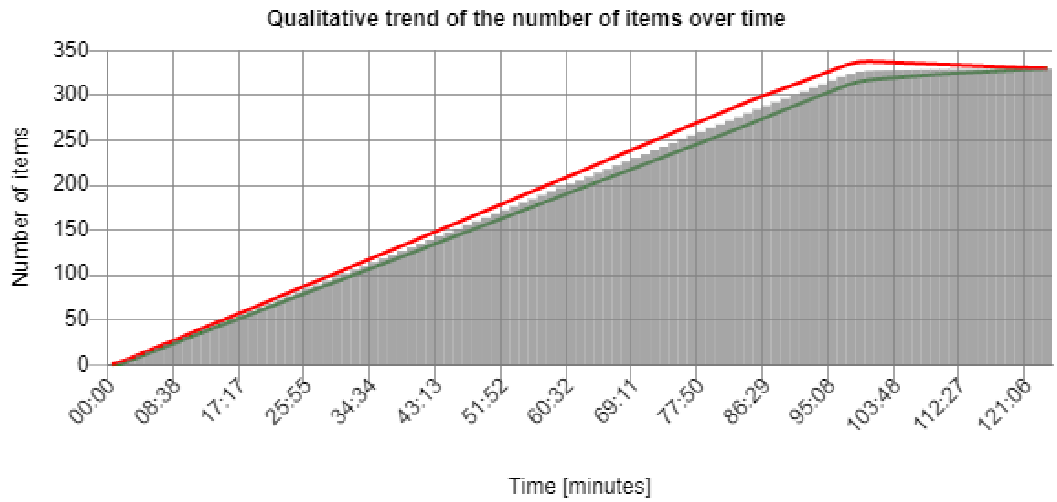

Figure 21: Graph of Warehouse

<u>N.B:</u> the number of elements in the warehouse is the number of pallets and not bottles.

The red and green lines in the graph represent the upper and the lower limit of the confidence interval, these intervals bound the results in the grey color

The following part includes tests similar to the previously performed ones related to buffer size reduction, parallel labelers and also additional tests on other elements that can be introduced into the production line.

Tests with mixed and combined ideas are also performed to check if better outputs can be obtained.

- **20% Buffer Size Reduction:**

The 20% reduction in the buffers sizes gives the results in Figure 22, Figure 23 and Figure 24 and takes 98min9sec to complete.

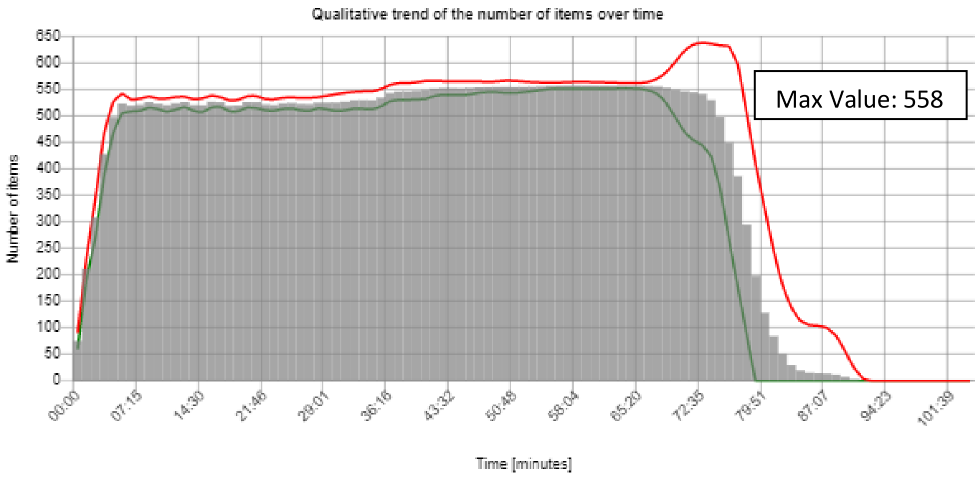

Figure 22: Graph of DepaToFill

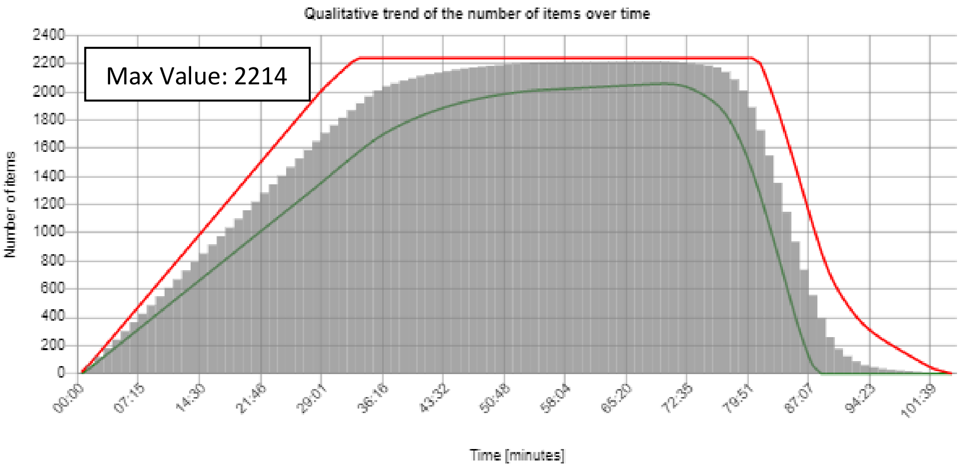

Figure 23: Graph of FillToLabel

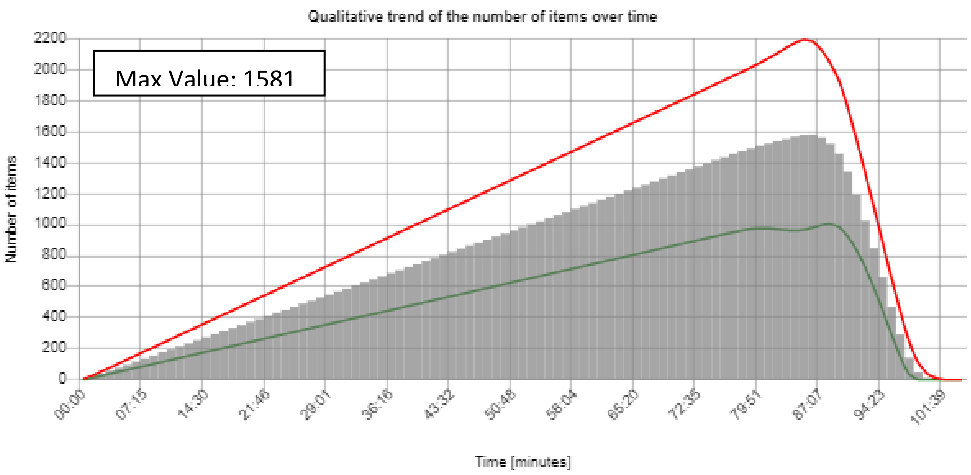

Figure 24: Graph of LabelToBox

Reducing buffers sizes for such a batch size does not affect the production time, instead it offers a reasonable safe change in the buffer sizes that can save valuable space in the production plant.

This reduction will only affect the production time in case where the buffers are saturated and machines would not take any new job, until their precedent buffer empties a space, this case will happen only with very large batches as shown in Table 13.

- **Parallel Machine:**

A Second Labeler machine added in parallel to the original one, having the same specifications, gave the results shown in Figure 25, Figure 26 and Figure 27, and took 97min10sec to complete.

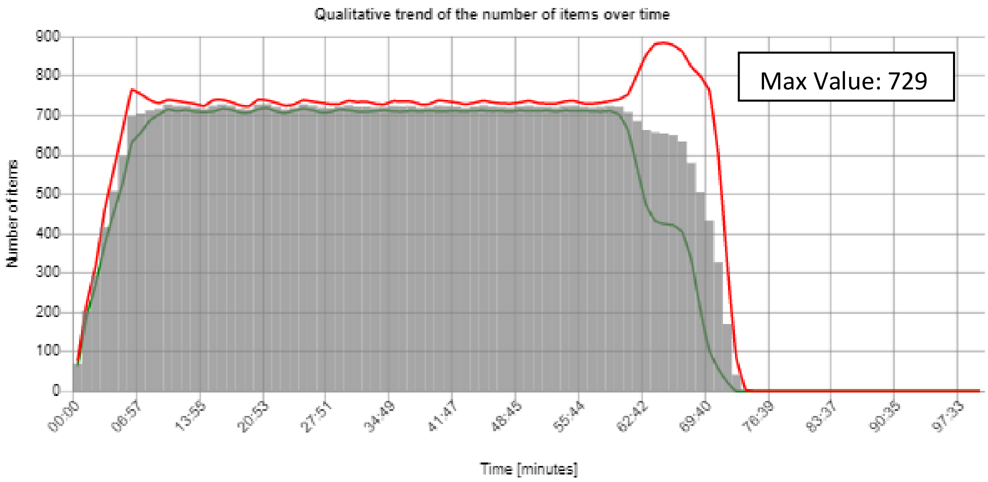

Figure 25: Graph of DepaToFill

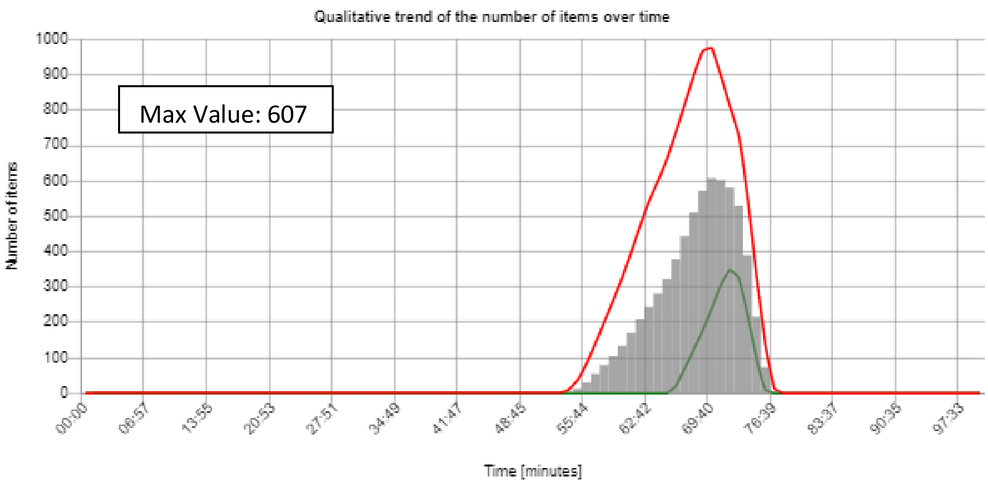

Figure 26: Graph of FillToLabel

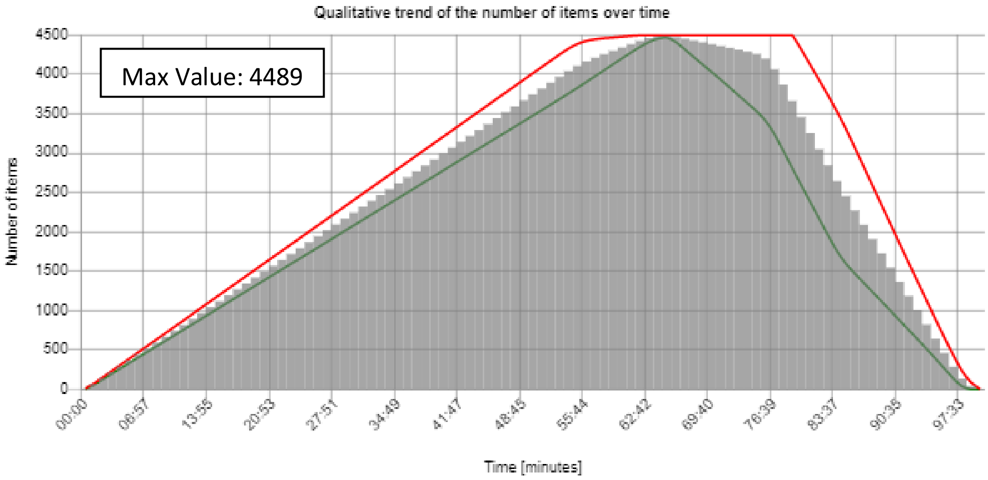

Figure 27: Graph of LabelToBox

Although adding a second Labeler in parallel did not reduce the production time by much, but it clearly affected the second and third buffers maximum value. It reduced the one related to the second buffer by more than 4 times, and almost totally filled the third buffer.

- **<u>Parallel Labeler with Extra Buffer between Boxer and Palletizer:</u>**

Since the third buffer is almost filled after adding the parallel Labeler, a simulation with an extra buffer added between the last two machines of the production line is done in order to distribute the bottles between them. This simulation took <u>83min57sec</u>.
All the results are shown in Figure 28, Figure 29, Figure 30 and Figure 31.

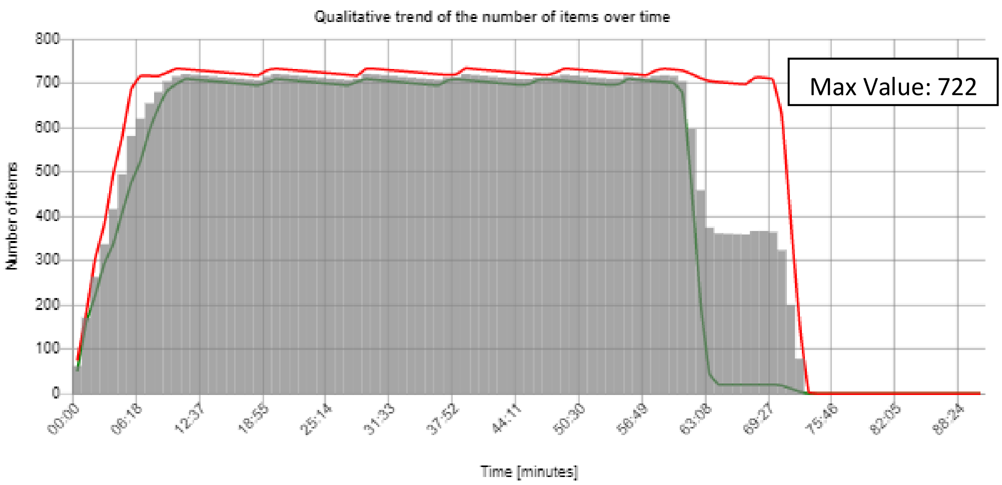

Figure 28: Graph of DepaToFill

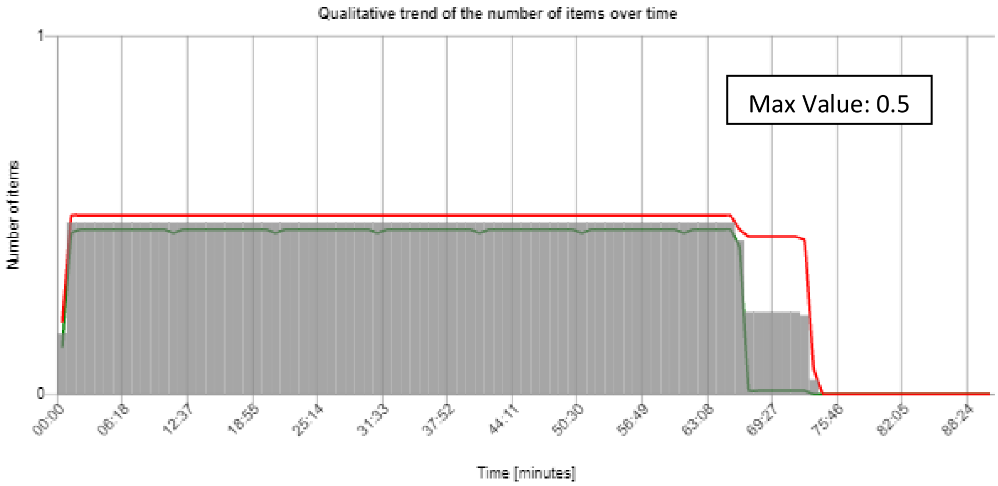

Figure 29: Graph of FillToLabel

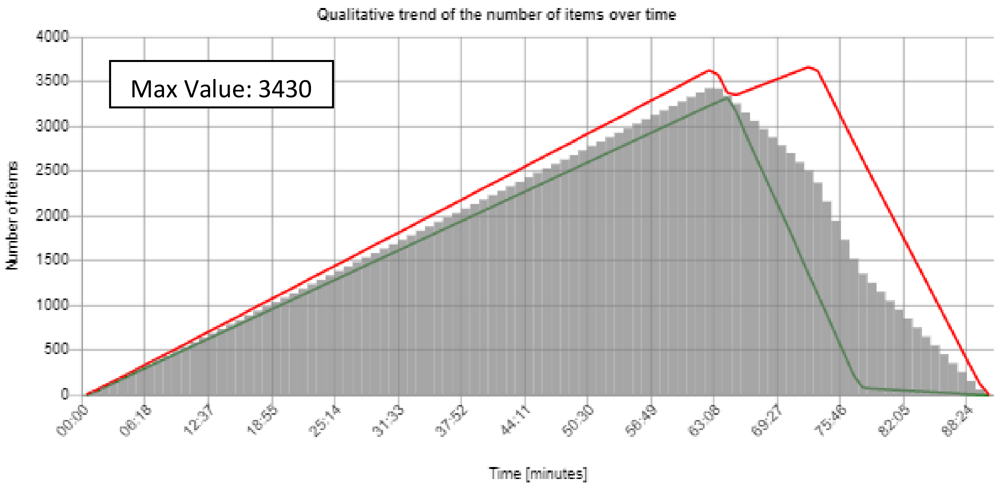

Figure 30: Graph of LabelToBox

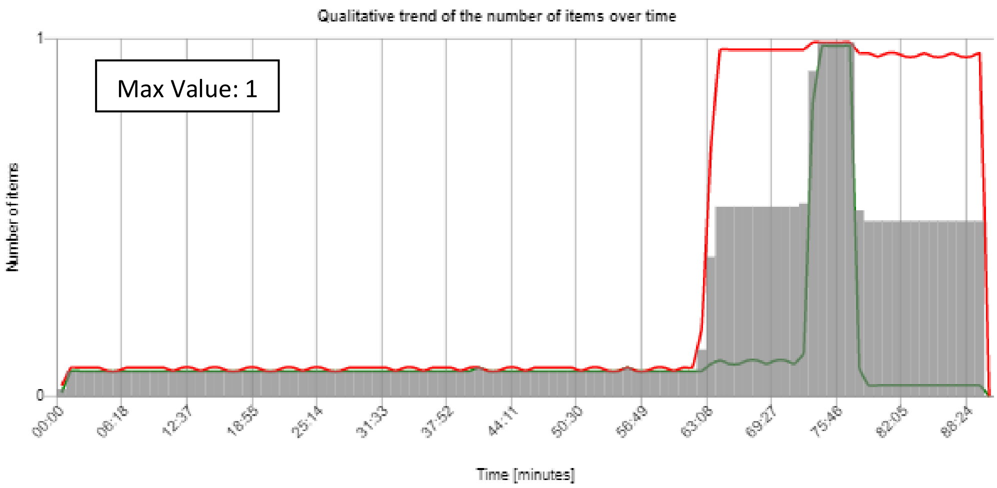

Figure 31: Graph of BoxToPal

This simulation has the best completion time achieved so far. In addition, it offers the chance to reduce the sizes of the buffers since none of them is completely filled now, which is a an optimization by itself.

- **Buffer Sizes Reduction, Parallel Labeler and Extra Buffer:**

As mentioned in the previous section, since none of the buffers are completely filled when the extra buffer is added, a 20% reduction in the sizes is performed. This simulation gives the best production line architecture, finding a trade-off between the production time (89min37sec) and the space used in the plant, since all buffer sizes are reduced.

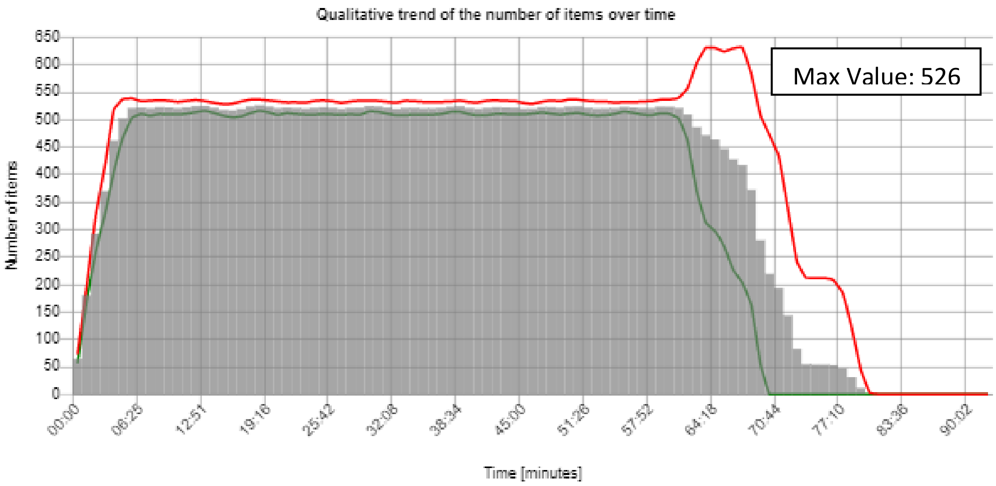

Figure 32: Graph of DepaToFill

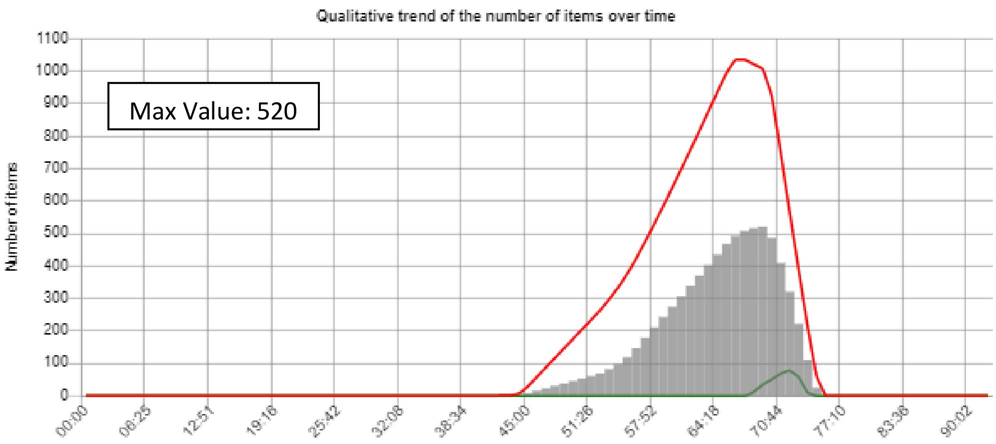

Figure 33: Graph of FillToLabel

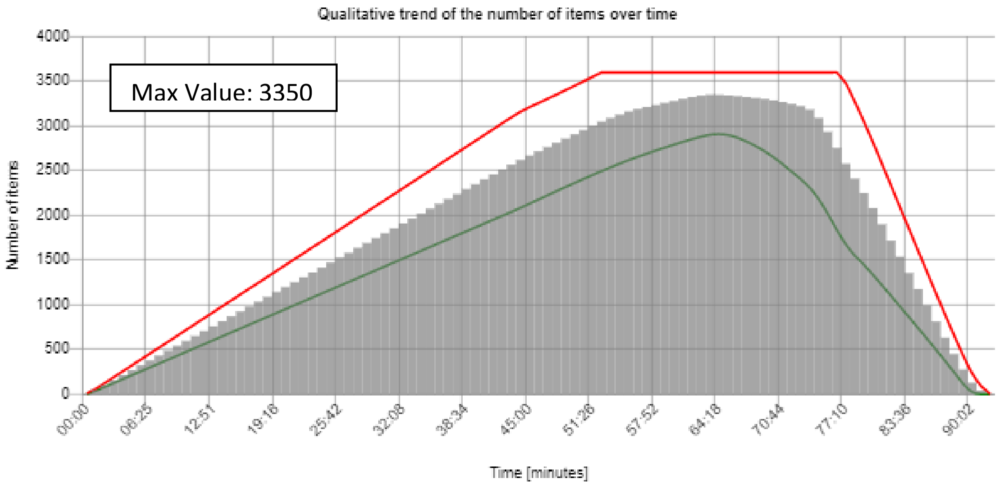

Figure 34: Graph of LabelToBox

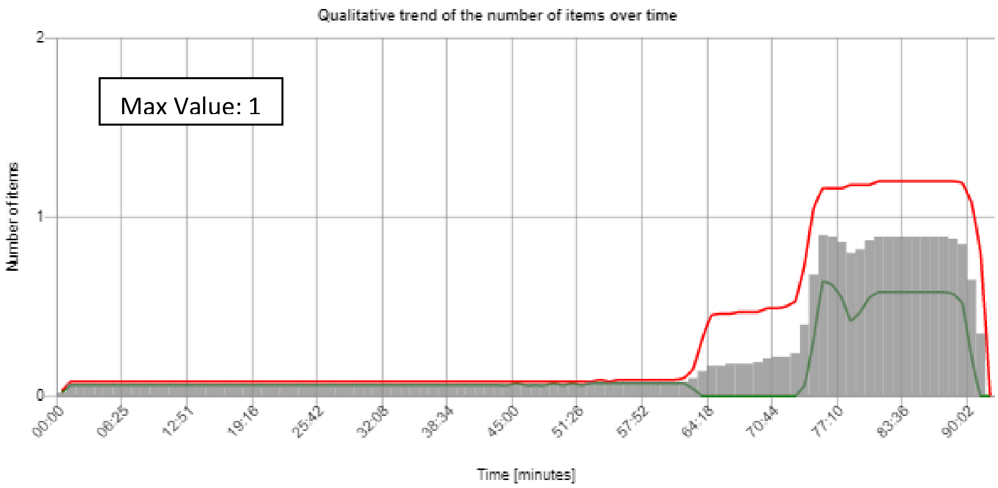

Figure 35: Graph of BoxToPal

It is appropriate to note that the extra buffer added can have a very low capacity as obvious in Figure 35, meaning such an addition can improve the line without much added expenses as long as the configuration of the line allows such an addition.

After performing the extra tests on the 20K batch, it is found that optimizations in the overall production time, and the space of the factory are possible. It is up to the company to study the actual money value of both factors, and decide on bases of what is more suitable for their own plans and interest and then act upon.

The tests can be recreated on different batch sizes, but due to very high simulation times, the provided samples are sufficient to understand the behavior of the process after each change.

The maximum value the simulator is able to handle is 80,000 after that the simulation times tend to increase dramatically and loops keep on running till the simulation is stopped by itself.

CHAPTER 4: CONCLUSION

The Optimization of a process is a simple goal, but the procedure is full of details and complex parameters that have mutual effect over other parameters, thus the studying and analysis in optimization processes is key to determine what is the best output that can be obtained. This is why, the studying in this thesis was separated so things can be handled in a precise and professional manner.

This thesis handled the first part of optimization related to the scheduling of the batches, in order to organize the weekly load of products in the most efficient way, saving as much time as possible. This was done using strategies and theories in the scheduling science and adapting the problem to the needed formats to allow the solving of such complex processes.

The scheduling is reached by feeding all the developed models of the problem to a solver that can run a very high number of iterations trying to eliminate the delays as much as possible.

The scheduling part is concluded with a complete, user friendly interface allowing the user to simply input the desired batches to be produced, and run the solver which then returns the organized optimized order of production with additional information related to the exact delays that will be induced.

This should help the working staff of any factory with mass production lines to insure that the scheduling of batches or jobs is efficient and time saving, and can be used as a mechanism to check the scheduling plans created by the people in charge of this procedure.

The second part of this thesis is dedicated to the optimization of the production line by studying optimization possibilities to reduce the overall simulation time, or reduce the size of the buffers used in the production process to save valuable space that can be used for other tasks.

The study focuses on performing simulations with the model of the production line and running multiple tests with different production configurations to insure that the data is true for all production sizes. The studies are documented and explained with analysis that allows the company to take decisions related to the suggested solutions.

The company can decide what is more worthy in terms of money worth, and act upon that decision by adding some elements to decrease the production process timing, or decreasing the buffer sizes to save space, and can also find other suggestions that can further more improve the architecture of the line.

In future work, the studies can be dedicated to finding the most appropriate reduction sizes that can be performed to the buffers, and other parts of the production process can be taken into consideration such as the Boxer machine that is the second bottle neck machine after the labeler.

References

Pinedo, M. (2002). *Scheduling-Theory, Algorithms and Systems* (Second ed.). Prentice-Hall Inc.

Xpress-Mosel Reference Manual. (2014). Fair Isaac Corporation, Release 3.8.

Xpress-Optimizer Reference Manual. (2017). Fair Isaac Corporation, Release 32.01.