Control of a complex high-mix-low-volume job shop manufacturing environment under disturbances

van Etten, B.J.

Award date:
2020

Department of Industrial Engineering and Innovation Sciences
Operations Planning Accounting and Control

Control of a complex high-mix-low-volume job shop manufacturing environment under disturbances

Master's Thesis

In partial fulfillment of the requirements for the degree of Master of Science in Operations Management and Logistics

B.J. (Bram) van Etten
Student Identity Number: 0866228
BSc in Industrial Engineering

University Supervisors
Dr. W.L. (Willem) van Jaarsveld, TU/e
Prof. Dr. A.G. (Ton) de Kok, TU/e
Dr. Ir. H.P.G. (Henny) van Ooijen, TU/e

Company Supervisor
A. (Arno) Vogels, VDL ETG Eindhoven

Eindhoven
May 20, 2020

Abstract

The effectiveness of controlled order release and machine dispatching (workload control) is a well-studied topic both in literature and in business. Work in progress reduction has been linked to an increase in effectiveness by reducing the mental strain on employees. In this thesis, we further contribute to the body of workload control literature by investigating the effectiveness of controlled order release and machine dispatching in reducing work in progress through a case study at a high-mix-low-volume job shop.

In a number of iterations we develop an extended job shop simulation model. This model includes the disturbing effects of yield loss, machine breakdowns, and external operations. We further include an overtime mechanism to model the aggregate effects of human intervention that are typically not considered in literature. This mechanism is triggered when the amount of arriving work or work in progress increases and results in a temporary increase in service rates. We demonstrate that inclusion of the disturbances and the overtime mechanism result in a model that is empirically valid, in the sense that typical simulations of the resulting model are in line with observations for a real-life job shop.

Using the developed simulation model we test the effectiveness of three known machine dispatching rules (earliest due date, least slack time, and critical ratio) and two order release rules (maximum number of jobs and path based bottleneck). Our tests show that work in progress reduction is linked with throughput reduction when the maximum number of jobs release mechanism is used, due to an increase in machine idleness. This decrease in throughput is almost completely prevented with the path based bottleneck release rule. For this rule large work in progress reductions are found with nearly no throughput loss, leading to a strong case for the adoption of such path based bottleneck release rules in real-life job shops.

Executive Summary

The dynamics of daily planning within the job shop of *Parts* are not fully understood which has led to little control on production. In an attempt to improve control, *Parts* increased safety times and started expediting delayed jobs by giving them priority over other jobs. The implementation of safety times increases work in progress inventory and the expediting of jobs creates nervousness. As a result, *Parts* finds itself in a downwards spiral with high work in progress inventory and low control on production. This high work in progress results in, a high working capital, ineffectiveness due to a full work floor, regular re-planning, and a high number of manual interventions. Also, research has shown that too high work in progress inventory leads to reduced effectiveness as a result of mental strain on employees. This means that a reduction in work in progress without a reduction in throughput would be beneficial for *Parts*. This research investigates how controlled order release and machine dispatching can be used to reduce the work in progress at *Parts* with little reduction in throughput.

To investigate the effectiveness of controlled order release and machine dispatching we build a simulation model representing the production activities of *Parts*. This model combines the standard job shop simulation model with extensions in the forms of yield loss, machine breakdowns, and external operations. In addition this model includes an overtime mechanism. This mechanism aims to capture all interventions done to increase productivity in case the available or planned work would exceed capacity by too much. The simulation model is empirically validated and used to test the performance of different order release and machine dispatching rules. In these tests, we change the dispatching and/or order release method in the model and compare the performance of the new model on three main KPIs in the form of work in progress reduction, throughput increase, and overtime reduction using the simulation model.

The three machine dispatching methods are earliest due date, least slack time, and critical ratio. For earliest due date and least slack time, we find an increase in throughput and a decrease in work in progress. This can be attributed to the fact that using these rules for machine dispatching will prioritize jobs with shorter processing times. However, continuously prioritizing these jobs might not be a suitable strategy if the profits of these jobs are included as well. For critical ratio, we also found an increase in throughput, however, this time paired with an increase in work in progress, which is undesirable. We conclude that although, smart machine dispatching can improve the performance of *Parts* the found benefits are small and this should not get priority at the moment.

The first order release rule tested is maximum number of jobs which limits the work in progress on the shop floor to a maximum. We found that this rule results in a decrease in work in progress paired with a rather substantial decrease in throughput. We conclude that this decrease in throughput is the result of machine idleness when work in progress at (temporary) bottlenecks stops order release. At *Parts* this effect is magnified due to the fact that jobs could be separated based on the type of starting material and the routings of these different types of jobs are unlikely to overlap significantly.

Aiming to reduce this idleness we test the effectiveness of the path based bottleneck release rule. The path based bottleneck release rule only releases those jobs for which releasing them will not exceed a load threshold for any machine in their routing. The results show that significant reductions in work in progress are possible with very small reductions in throughput. Considering that past research showed that decreasing the work in progress can lead to increases in effectiveness we believe that this reduction in throughput will be nullified or even changed into an increase.

Implementing a pre-shop pool will lead to a very substantial reduction in work in progress inventory on the shop floor, and thus a reduction in working capital. Moreover, by combining our insights with literature we may expect that a pre-shop pool will increase the throughput of the shop, reduce the number of manual interventions once parts are admitted on the shop floor, and improve worker satisfaction and retention.

Preface

This thesis is not only the result of seven months of conducting research. It is also the final requirement of my master's program and thus marks the end of my six and a half-year student life. This amazing period had very different phases from each of which I have great memories and even better friends. From enjoying the freedom of student life and all the parties that come with it in my bachelor's, riding the roller coaster that is a board year of Industria, further developing my academic skills during my master's, exploring Taiwan's amazing sceneries and people as an exchange student at the Taiwan National University, and finally tackling a real-life business problem at VDL ETG Eindhoven in this project. I would like to use this opportunity to thank some of the people who were part of this wonderful journey.

First of all, I would like to thank my amazing parents. Thank you for your unconditional love, for supporting me in everything I do and for teaching me to always create my own values and opinions. I would like to thank Joost and Jaap for being awesome brothers and friends. Jaap, I would like to also express my gratitude for all the valuable feedback you gave me on this project. I want to thank my girlfriend Floor for her love and support. It is amazing to create so many beautiful memories together, and I'm looking forward to create many more. I want to thank my friends from the Reyneart College, for all the laughter and joy. Special thanks goes out to Sjoerd for being the best friend anyone could wish for and for being there for me where-ever I am and what-ever I do. Thank you to all the fantastic people I met in Eindhoven, specifically the members of the 53th board of Industria for the unforgettable year that we experienced intensively together.

Then there are several people from the TU/e whom I would like to thank. I am grateful for the guidance I have received from Willem van Jaarsveld as mentor during my master's and this project in particular. Willem, you always made it clear this was our project and we were going to tackle the problems, which made your recommendations that much more valuable to me. I enjoyed our shared astonishment on what we discovered during the project and the breakthroughs we made during the many discussions we had. Also, I would like to thank Ton de Kok for sharing his incredible knowledge and experience both in research and tackling business problems. I want to thank Henny van Ooijen as well for taking the time to be the third assessor of my work.

I am thankful for the people who made my research at VDL ETG Eindhoven possible. Starting with Arno Vogels. Arno, your experience as a supply chain engineer, knowledge on the dynamics within the company, and positive attitude towards the project were all very valuable. Hein Brughmans, thank you for our weekly meetings and advice on how to keep the project relevant for both the company and the university. Finally, special thanks go out to Aroop Bhattacharjee. Aroop, thank you for all the effort you put in making me feel at home, all the brainstorm sessions and discussions we had, but most of all for the occasional life lessons.

I am glad I have been able to apply much of the knowledge and experiences that I have gained at the TU/e, both during my bachelor's and master's, to this thesis. Even though it feels like there will always be more I would like to add and aspects I would like to improve on, I am proud of the end result and hope you will enjoy reading it.

Bram van Etten

Contents

Contents iv

List of Figures vi

List of Tables vii

List of Algorithms viii

List of Abbreviations ix

1 Introduction 1

1.1 Company Description 1
1.2 Outline 1

I Research Context 2

2 Research Design 3

2.1 Problem Formulation 3
2.2 Research Questions 7
2.3 Research Framework 7

3 Literature Review 9

3.1 Order Review and Release 9
3.2 Shop Floor Control 11
3.3 Conclusion 13

II Model development 14

4 Conceptual Model 15

4.1 Structure 15
4.2 Assumptions 16
4.3 Validation 16

5 Computerized Model 17

5.1 Objects 17
5.2 Simulation 18
5.3 Events 19
5.4 Distributions 24
5.5 Rules 26
5.6 Monitoring 27
5.7 Conclusion 27

6 Validation and Verification 28

6.1 Computerized Model Verification 28
6.2 Operational Validation 30
6.3 Summary 32

III Solution design **33**

7 Results **34**
7.1 Key performance indicators . . . 34
7.2 Order release and machine dispatching . . . 34
7.3 Summary . . . 43

8 Conclusion **44**
8.1 Conclusions . . . 44
8.2 Reflection . . . 45
8.3 Recommendations . . . 46

References **48**

Appendix **52**

A Overview of sequence and release rules **52**
A.1 Overview of notation . . . 52
A.2 Overview of sequence rules . . . 53
A.3 Overview of release rules and their classification . . . 54

B Conceptual model **55**
B.1 Conceptual model systematic representation . . . 55
B.2 Conceptual model assumptions . . . 55

C Overview of events **56**

D Data sets **57**
D.1 Data collection . . . 57
D.2 Data quality . . . 57
D.3 Data inclusion rules . . . 58
D.4 Data consolidation . . . 58

E Overview of simulation parameters **59**
E.1 Distributions used . . . 59
E.2 Overall parameters . . . 59
E.3 Priority parameters . . . 59
E.4 Due date parameters . . . 60
E.5 Machine parameters . . . 61

F Validation and verification tables and overviews **62**
F.1 Overview of face validation relations . . . 62
F.2 Comparison to queuing models . . . 63
F.3 Comparison to Jackson Networks . . . 63
F.4 Comparison to yield formulas . . . 64

G Model sensitivity **65**
G.1 Model sensitivity with respect to work in progress . . . 65
G.2 Model sensitivity with respect to effective yield . . . 65

List of Figures

2.1 Delivery performance in percentage of jobs not tardy per week at *Parts* 3
2.2 Pareto of planned cycle efficiency at *Parts* 3
2.3 Average work in progress per week in months time supply at *Parts* 4
2.4 Backlog per machine station in weeks at *Parts* 4
2.5 Modified response time spiral adapted from Suri (1998) 4
2.6 Systematic representation of relationship between throughput KPIs adapted from Germs (2012) 5
2.7 Performance Arousal Curve adapted from Diamond et al. (2007) 6
2.8 Job demands-resources model adapted from Demerouti and Bakker (2011) 6
2.9 Work efficiency smiley adopted from Zwartelé (2016) 6
2.10 Research design 8

3.1 Simplified Model Building Process adopted from Sargent (1981) 14

4.1 Conceptual Job Shop Model 15

7.1 Path based bottleneck with critical ratio dispatching for different threshold values 42
7.2 Path based bottleneck with least slack dispatching for different threshold values 42

B.1 Conceptual Job Shop Model 55

D.1 Data inclusion rules 58

F.1 Structure of Jackson Network for comparison 63

List of Tables

3.1 Classification framework for Order Release adapted from Bergamaschi et al. (1997) 11

5.1 Overview of job attributes 17
5.2 Overview of job states 17
5.3 Overview of machine attributes 18
5.4 Overview of machine states 18
5.5 Overview of pre-shop pool states 18
5.6 Distribution of job priorities 25

6.1 Overview of model iterations 32

7.1 Immediate Release 37
7.2 Maximum Number of Jobs - Work in progress reduction 38
7.3 Maximum Number of Jobs - Throughput increase 38
7.4 Maximum Number of Jobs - Overtime decrease 39
7.5 Maximum Number of Jobs - Effective throughput increase 39
7.6 Path Based Bottleneck 41
7.7 Path Based Bottleneck with increased efficiency 42

A.1 Overview of sequence rules 53
A.2 Release rules and their classification 54

C.1 Overview of events 56

D.1 Overview of data sets 57
D.2 Example data finished operations 57

E.1 Overview of distributions used 59
E.2 Overview of overall simulation parameters 59
E.3 Distribution of job priorities 59
E.4 Distribution of job priorities 60
E.5 Machine process times 61

F.1 Comparison to m/m/1 queue 63
F.2 Comparison to m/m/1 priority queue 63
F.3 Comparison to Jackson Network 64
F.4 Comparison to extended yield formula 64

G.1 Model sensitivity with respect to work in progress 65
G.2 Model sensitivity with respect to effective throughput 65

List of Algorithms

5.1 Simulation . . . 19
5.2 Order Arrival . . . 20
5.3 Order Release . . . 20
5.4 Next Operation . . . 20
5.5 Check Overtime Policy . . . 21
5.6 Start Operation . . . 22
5.7 External . . . 22
5.8 End Processing . . . 23
5.9 Repaired . . . 23
5.10 Rework . . . 24
5.11 First Come First Serve . . . 27
5.12 Immediate release . . . 27

7.1 Earliest due date . . . 35
7.2 Least Slack Time . . . 35
7.3 Critical Ratio . . . 36
7.4 Maximum number of jobs . . . 37
7.5 Path based bottleneck . . . 40
7.6 Slack Ratio . . . 41

List of Abbreviations

BFL	Backward finite loading
BIL	Backward infinite loading release rule
CSCOR	Capacity slack corrected sequence rule
DLR	Due date and load-based release rule
EDD	Earliest due date sequence rule
ERD	Earliest release date sequence rule
ETG	VDL ETG Eindhoven
FCF	First come first serve equence rule
FFL	Finite forward loading release rule
HOR	Highest number of operations remaining sequence rule
IMM	Immediate release release rule
JD-R	Job-demands-resources
KPI	Key Performance Indicators
LAD	Latest arrival date sequence rule
LAJD	Look ahead job demanding sequence rule
LIF	Last in first out sequence rule
LOR	Least number of operations remaining sequence rule
LOS	Least operational slack time sequence rule
LPT	Longest processing time sequence rule
LST	Least slack time sequence rule
LWINQ	Least work in next queue sequence rule
LWKR	Least work remaining sequence rule
MCR	Minimum critical ratio sequence rule
MIL	Modified inifinite loading release rule
MJT	Maximum job tardiness sequence rule
MNJ	Maximum number of jobs release rule
MOD	Modified operational due date sequence rule
MODCR	Modified critical ratio sequence rule
MODCS	Modified capacity slack sequence rule
MSR	Minimum slack ratio sequence rule
MWKR	Maximum work remaining sequence rule
NINQ	Number in next queue sequence rule

OCR	Operational critical ratio sequence rule
ODD	Operational due-date sequence rule
OEM	original equipment manufacturers
OR	Order release
ORR	Order review and release
PBB	Path based bottleneck
PIOC	Planned input-output control release rule
PSP	Pre-shop pool
PTT	Pre-shop pool Throughput Time
RQ	Research question
RTS	Response time spiral
SKU	Stock keeping unit
SPO	Slack per remaining operation sequence rule
SPT	Shortest processing time sequence rule
SRQ	Sub-research question
STT	Shop floor Throughput Time
TSPT	Truncated shortest processing time sequence rule
TTT	Total Throughput Time
WIP	work in progress inventory

1 Introduction

This report presents the results of a master thesis project conducted by the Eindhoven University of Technology in collaboration with VDL ETG Eindhoven (hereafter ETG). ETG is a supplier of original equipment manufacturers (OEMs) and strives to get a better understanding into the effects of internal decisions on their production process. We specifically focus on controlled order release and machine dispatching (workload control).

1.1 Company Description

ETG consists of multiple business units, the most important two are *Parts* and *Systems*. *Parts* is an internal supplier for *Systems* and is a job shop, whereas *Systems* is responsible for assembly. This research focuses on *Parts*. To get a better understanding of the role of *Parts* within ETG's supply chain, planning and control in the supply chain is discussed. Within the supply chain components are either produced internally at *Parts* or purchased directly from the supplier. *Parts* is organised as a job shop with roughly fifty different machines operated by specialists. Each specialist is trained to operate one or multiple machines. *Parts* receives the raw materials directly from suppliers. Within *Parts* the planning is done by *planning parts* which receives planning suggestions from *integral planning*. The components, either produced at *Parts* or purchased directly from the supplier, go to *Systems*. Within *Systems* assembly of components is done based on the planning of a *production assistant* who similarly to *planning parts* receives planning suggestions from *integral planning*. Finished systems are sent to *Expedition* which, based on information from *order management*, fulfils shipment to the customer.

Parts operates as a job shop producing a high-mix of products with low volumes. Therefore, *Parts* is classified as a high-mix-low-volume job shop. Within this job shop jobs arrive as production orders. Each job is for the production of one or more of the same stock-keeping units (SKUs). The production of a job generally consists of multiple tasks which have to be completed on different machines. Information on what tasks have to be completed in which sequence is stored in the job's routing. The regular flow of a job is as follows, first, a production map is made, whereafter it is checked whether the materials needed for production are available. When this is the case the job enters the queue for the first task in the routing. On completion of a task, the job enters the queue for the next task in the routing until the final task is done. Finally, the job is send to the finished components stock which marks the end of the process for *Parts*.

1.2 Outline

This remainder of this thesis is divided into three parts: Part I discussing the context of the problem discussed in this research, Part II introducing the model developed for analysis, and Part III analysing possible solutions. Part I starts by introducing the current situation at *Parts*, formulating the problem statement, and introducing the research framework used throughout this thesis in Chapter 2. Thereafter, we present a survey of relevant literature in Chapter 3. Part II concerns the introduction of the model. First, Chapter 4 introduces the conceptual model. Next, Chapter 5 introduces the computerized model. This model is a discrete event simulation model and the result of multiple iterations of model building. Chapter 6 concludes Part II with a discussion of validation and verification of the computerized model. This thesis continues in Part III with the analysis of workload control using the model build in Part II. In Chapter 7 we present possible methods for order release and shop floor control and test these methods using our simulation model. Chapter 8 continues by drawing conclusions based on these results. In addition to these conclusions we reflect on our research, and end this thesis with recommendations for *Parts*.

Part I
Research Context

In this Part, we introduce the context of the problem under investigation in this thesis. We start with the current situation at *Parts* and investigate the disturbances in the production process. Afterwards, we discuss how these disturbances arise and how *Parts* is trying to cope with them. Moreover, it is explained what we believe to be the main disturbing factor of the production process. Based on this we formulate a problem statement and corresponding research questions. Moreover, we review existing literature on job shop control to identify workload control techniques that could be useful in tackling this problem.

This Part is organised as follows. Starting in Chapter 2, we first formulate the research problem in Section 2.1, the research questions answered in this thesis are formulated in Section 2.2, and the framework used for answering these questions in Section 2.3. Thereafter, in Chapter 3 we present a survey on literature on order review and release in Section 3.1 and shop floor control in Section 3.2.

2 Research Design

In this chapter, we first formulate the research problem in Section 2.1. The research questions answered in this thesis are discussed in Section 2.2. Finally, Section 2.3 discusses the framework used for answering these questions.

2.1 Problem Formulation

Within *Parts* the dynamics of daily planning are not fully understood and there is little control on production. This little control and several disturbances lead to nervousness negatively influencing the production process. Slightly modifying the definition of Fadil et al. (1997), this research defines a disturbance as: "every event which can harm the work of a work centre, for which control is already specified". This research defines control as the process of releasing and sequencing jobs to streamline production. This section discusses how these disturbances arise and how *Parts* is trying to cope with them. Moreover, it is explained what we believe to be the main disturbing factor of the production process.

2.1.1 Effects of disturbances

As mentioned there is little control within *Parts*. This little control resulted in a low delivery performance as can be seen in Figure 2.1. To cope with the disturbances and the low delivery performance, safety times are implemented, leading to an increase in waiting time. Moreover, for each task, a predetermined amount of waiting days has been implemented in a response to the low delivery performance. As a result, jobs have high cycle times consisting mostly of waiting. Figure 2.2 shows a Pareto of jobs based on their planned cycle efficiency. As illustrated in this figure, more than 80% of the jobs have a planned cycle efficiency below 15%. In other words, for over 80% of the jobs it is planned that these jobs are waiting more than five times as long as they are in production.

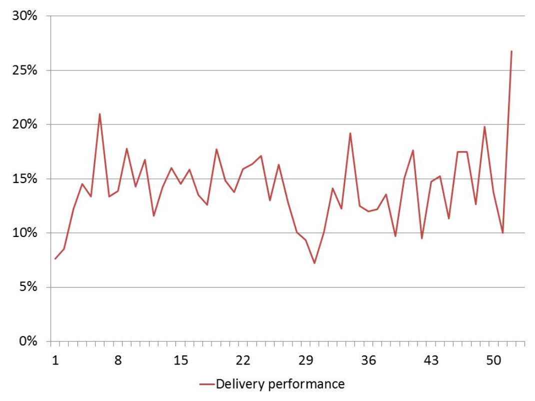

Figure 2.1: Delivery performance in percentage of jobs not tardy per week at *Parts*

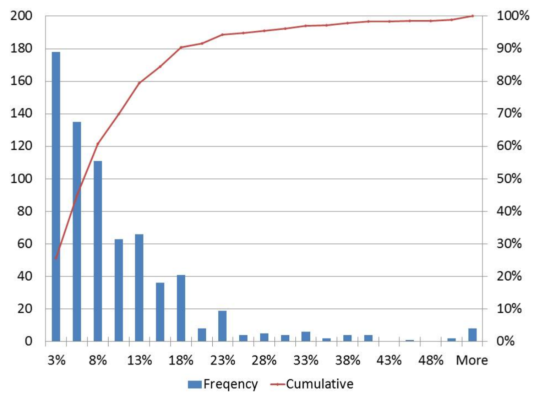

Figure 2.2: Pareto of planned cycle efficiency at *Parts*

These high waiting times have led to a work in progress inventory (WIP) equal to four and a half months worth of time supply. Figure 2.3 shows how this WIP developed over the past three years. Subsection 2.1.2 discusses the effects of this high WIP in more detail. To ensure timely delivery despite this high WIP, *Parts* implemented a priority system in which late jobs get expedited. At the moment, 32% of the orders are classified into one of four priority levels, resulting in the delay of the other 68% of jobs.

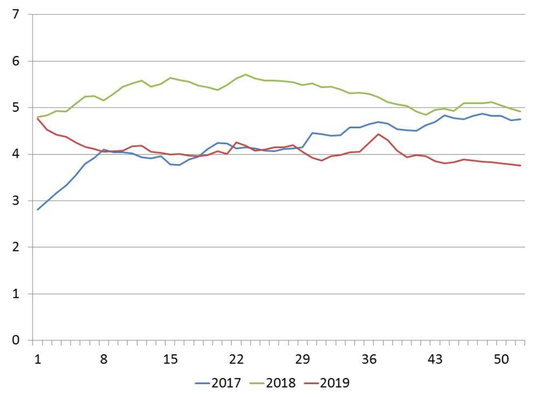

Figure 2.3: Average work in progress per week in months time supply at *Parts*

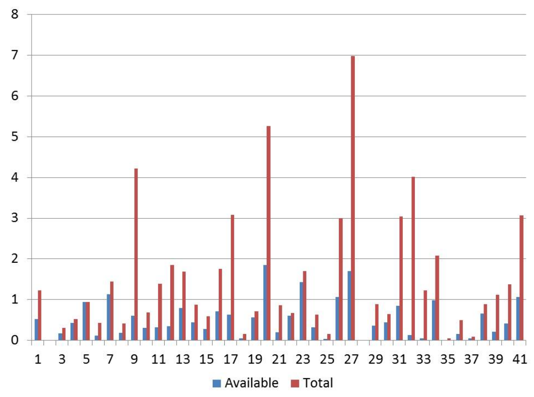

Figure 2.4: Backlog per machine station in weeks at *Parts*

The emerging nervousness from the expediting and delaying of jobs adds to the already existing nervousness. The resulting backlogs are high (Figure 2.4) and there is even lower control on production. This creates a downward spiral resembling the response time spiral (RTS) introduced by Suri (1998). Figure 2.5 shows a modified version of the RTS applicable to the situation at *Parts*.

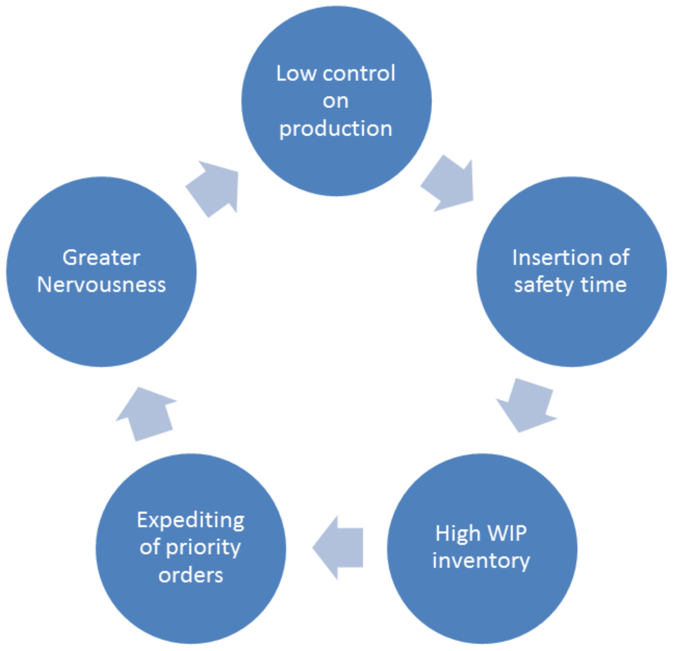

Figure 2.5: Modified response time spiral adapted from Suri (1998)

2.1.2 Effects of high work in progress inventory

In recent years the relationship between work in progress and delivery performance has been a topic of research. There appears to be a discrepancy between theoretical and empirical studies on this matter. Van Ooijen (1996) argued that by restricting the work in progress based on the capacity utilisation of machines, the delivery performance could be improved. This has been confirmed by earlier empirical studies

(Bertrand and Wortmann, 1981; Fry and Smith, 1987). Moreover, a study into many different control policies suggests that a reduction in Shop Floor Throughput time (STT) can be achieved by reducing WIP (Bertrand and Van Ooijen, 2002). Note that SST is the time between a job's release and its departure and thus does not include its pre-shop pool throughput time (PTT). The Total Throughput Time (TTT) is the time between arrival and departure of a job and therefore the sum of STT and PTT. Figure 2.6 gives a visual overview of the types of throughput. The reduction in STT found in these studies is offset by an increase in PTT, often leading to an increase in TTT. Based on this information, we conclude there is a discrepancy between the empirical results, suggesting the control of workload can decrease TTT, and theoretical studies showing an increase in TTT as a result of workload control.

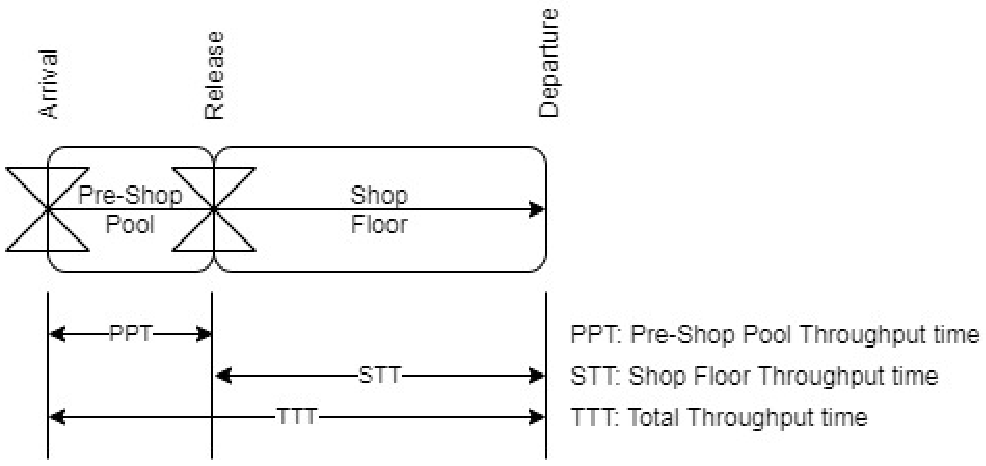
Arrival
Release
Departure
Pre-Shop Pool
Shop Floor
PPT
STT
TTT
PPT: Pre-Shop Pool Throughput time
STT: Shop Floor Throughput time
TTT: Total Throughput time

Figure 2.6: Systematic representation of relationship between throughput KPIs adapted from Germs (2012)

Bertrand and Van Ooijen (2002) give an explanation for this discrepancy. They suggest that workstations lose efficiency if the workload at that workstation is different from some optimal workload. This loss of efficiency is consistent with the performance arousal curve proposed by Yerkes and Dodson (1908) and shown in Figure 2.7. The performance arousal curve states that there is an inverted u-shape relationship between performance and arousal when completing difficult tasks. Arousal can be caused by stressors like the number of jobs to be completed at a certain work station. Thus, if we assume the tasks operators have to complete at machines to be difficult, there is an inverted u-shape with the performance of these operators and the number of jobs to be completed.

More support for the fact that increasing the WIP at a workstation could negatively influence the performance at that workstation is found in the form of the job-demands-resources (JD-R) model (Demerouti and Bakker, 2011). Similar to the performance arousal curve, this model links stressors (job demands) to performance (organizational outcomes). This model also includes a buffering effect of job resources, meaning that the effects of high job demands could be offset by boosting motivation and energy levels by adding resources. The JD-R model thus states that reducing job demands while keeping job resources the same will result in better performance. The performance arousal curve and the JD-R model both give a psychological explanation that a reduction in WIP can lead to a increase in performance.

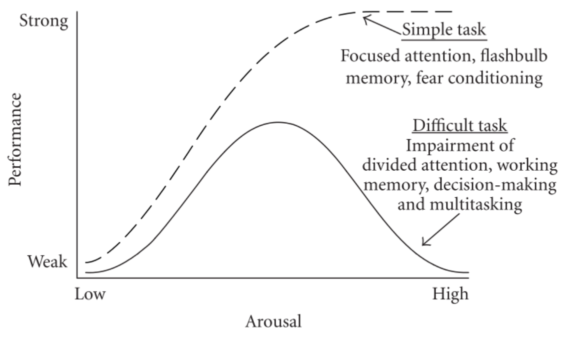

Figure 2.7: Performance Arousal Curve adapted from Diamond et al. (2007)

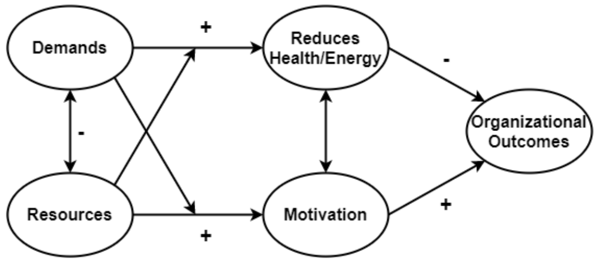

Figure 2.8: Job demands-resources model adapted from Demerouti and Bakker (2011)

In addition to the psychological explanation given by the performance arousal curve and the JD-R model there is a mathematical model explaining how a reduction in WIP could be beneficial. Little's Law states that the average number of jobs in a stable system (WIP) is equal to the average number of job arrivals to the system (λ) times the average time a job spends in the system (TTT)(Whitt, 1992). Equation 1 shows Little's Law formally. Little's Law shows that as long as the system is stable and the arrival rate constant, a reduction in WIP would lead to a proportional reduction in TTT and thus increase the performance. While Little's Law shows that a reduction in WIP would lead to a reduction in TTT. A reduction in WIP can also increase the systems idleness and thus reducing the total throughput of a system. This again suggests an inverted u-shape relation between WIP and performance.

$$WIP = \lambda * TTT \tag{1}$$

For *Parts* the high WIP means a high working capital. An additional consequence of the high WIP, are long and uncertain lead times, leading to regular re-planning (often multiple times for the same order) of jobs. In the process of re-planning a new expected delivery date is communicated to the customer. Moreover, the high WIP leads to full workstations and frequent transporting from and to warehouses all decreasing efficiency.

The workload efficiency smiley, illustrated in Figure 2.9, shows all the aforementioned relationships between WIP and performance in a simple way. From this model we can conclude that if it is possible to reduce the WIP with little effect to other KPIs, the performance would increase.

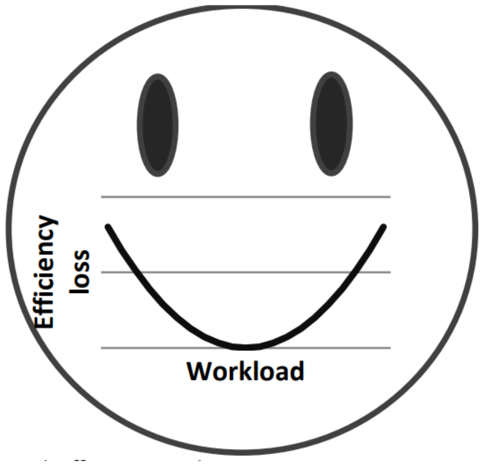

Figure 2.9: Work efficiency smiley adopted from Zwartelé (2016)

2.1.3 Problem statement

In Sections 2.1.1 and 2.1.2 we have shown how disturbances lead to high WIP at *Parts*. Furthermore, we explained how this high WIP leads to an even higher WIP following an adjusted version of the RTS. Finally, we discussed the negative effects of this high WIP and the possible benefits of reducing it. This can be summarized in the following problem statement:

The effects of disturbances on production lead to high work in progress inventory. It seems that the current methods of coping with this high WIP lead to an even higher WIP. As a result, there is little control on production leading to frequent late deliveries.

As discussed, we believe reducing the WIP can break the RTS and thus improve the control of *Parts*. However, to be beneficial, the reduction in WIP should not lead to high reductions in throughput. Section 2.2 formulates the research questions answered in this thesis.

2.2 Research Questions

Based on the problem statement given in Section 2.1 the following main research question (RQ) is formulated:

How to control order release and dispatching in a complex high-mix-low-volume job shop manufacturing environment to reduce work in progress inventory without reducing throughput?

To answer the main RQ it is divided into three sub-research questions (SRQs). Firstly, SRQ1 focuses on model development. SRQ2 and SRQ3 use the developed model to generate quantitative insights. The proposed research aims to model production within *Parts* through simulation. Therefore the first sub-research question follows logically:

1. *How can a complex high-mix-low-volume job shop manufacturing environment be simulated?*

After the simulation model is built it will be used to investigate what effect dispatching has on the control of the system. Effectively this means, testing whether dispatching can reduce the WIP without disrupting the throughput, resulting in RQ2. Similarly, the simulation model will be used to investigate the effect of order release on the WIP and throughput of the system. RQ3 is formulated to answer this.

2. *How can dispatching improve control of a complex high-mix-low-volume job shop manufacturing environment by reducing work in progress inventory without reducing throughput?*

3. *How can order release improve control of a complex high-mix-low-volume job shop manufacturing environment by reducing work in progress inventory without reducing throughput?*

The remainder of this thesis is used to answer these research questions. Section 2.3 describes how the research is set up to achieve this. Moreover, in Chapter 8 these research questions will be recapped and their answers will be discussed.

2.3 Research Framework

To be able to answer the RQs as defined in Section 2.2 a research is designed as outlined in this chapter. Figure 2.10 gives a systematic representation of the research design. As illustrated, a simulation model, able to represent production within *Parts*, will be made. This modelling of the real world situation corresponds to RQ1. Next, the effects of order release rules and dispatching rules will be tested using the simulation model build to answer RQ2 and RQ3. Finally, all acquired insights will be combined into conclusions, recommendations, and further research directions.

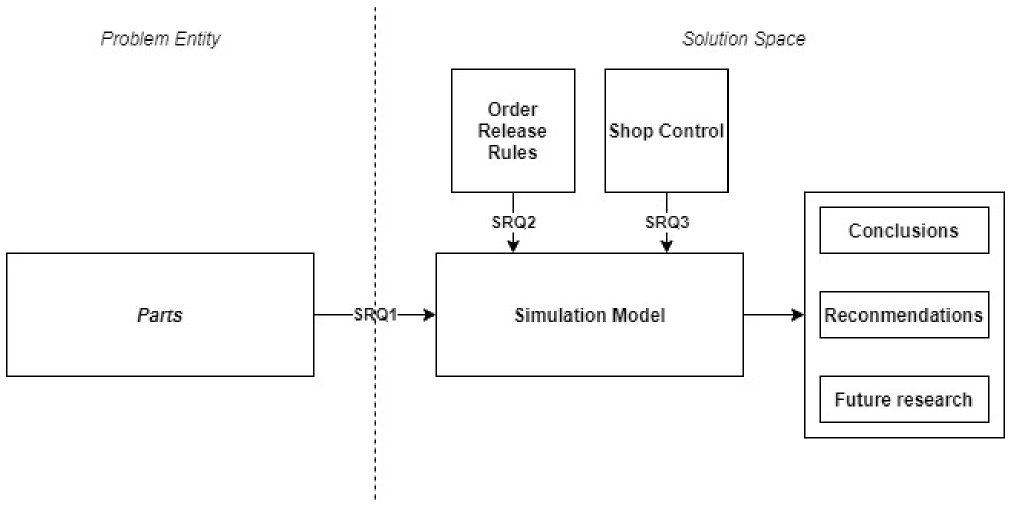

Figure 2.10: Research design

3 Literature Review

In this chapter we review existing literature on job shop control. Job shop control can be classified into three different categories: *Order Entry*, *Order Review and Release*, and *Shop Floor Control*. *Order Entry* concerns the decision to accept or decline a job, *Order Review and Release* manages the transition of jobs from planning to production, and *Shop Floor Control* controls the sequence in which jobs are processed (Bergamaschi et al., 1997). Order Entry is out of scope of this thesis and is therefore excluded from this literature review. Hence, the remainder of this chapter focuses on *Order Review and Release* and *Shop Floor Control* which will be discussed in Sections 3.1 and 3.2 respectively.

3.1 Order Review and Release

In this section, an overview of *Order Review and Release* (ORR) methods in the literature is given. The objective of ORR is to manage the transition of jobs from planning to production (Bergamaschi et al., 1997). ORR consists of two main phases, the pre-shop pool (PSP) phase, and the order release (OR) phase. In the PSP phase the jobs are queued in the PSP in according to some sequence rule. For a complete overview of sequence rules discussed in this thesis see Appendix A.2. In the OR phase jobs are released at specific times and specific quantities based on release rules. The remainder of this section is structured as followed. First, an overview of pre-shop pool sequence rules if given in Subsection 3.1.1. Next, Subsection 3.1.2 gives an overview of release rules. Finally, Subsection 3.1.3 gives an classification of release rules.

3.1.1 Pre-Shop Pool phase

The most basic PSP load based queuing rule introduced is *first come first serve* (FCF) (Fredendall et al., 2010). FCF simply queues all jobs based on the their time of arrival, with the job with the earliest arrival time first. In contrast the *earliest due date* (EDD) rule sequences jobs based on their due-date with, as the name suggests, the earliest first. This rule has among others been discussed in pre-shop pooling literature by Ragatz and Mabert (1988); Melnyk and Ragatz (1989); Philipoom and Fry (1999); Philipoom and Steele (2011). Bechte (1988) proposed to use the *earliest release date* (ERD), sometimes referred to as *planned release date*, as sequencing rule for PSP. ERD sequences the jobs based on their planned release date with the earliest planned release date first (Land and Gaalman, 1998; Perona and Portioli, 1996; Fredendall et al., 2010; Thürer et al., 2011, 2012).

To balance between the due date and the processing time the *critical ratio* (MCR) rule was introduced. MCR sorts the jobs based on increasing critical ratios. The critical ratio of an order is equal to the time remaining until its due date divided by the remaining processing time (Bobrowski, 1989; Abu-Suleiman et al., 2005). A system in which the PSP is sequenced according to the slack ratio each job has, *minimum slack ratio* (MSR), was proposed by Philipoom et al. (1993) and later used by several others (Fredendall et al., 2010; Malhotra et al., 1994). The slack ratio of a job is defined as the proportion of machine slack that would be consumed by the release of the job, where machine slack is the difference between a predefined threshold and the current work already committed to the machine. To correct the load of a job for the time the job is actually at a machine Thürer et al. (2015) proposed the *capacity slack corrected* rule (CSCOR). The *modified capacity slack* rule (MODCS) is a combination of ERD and CSCOR (Thürer et al., 2015). MODCS first sorts all jobs in the PSP into two category levels, one for urgent jobs, (i.e. jobs with a release date within of before the current release period) and regular jobs (all other jobs). Following MODCS the urgent jobs always get priority over the regular jobs, consecutively the urgent jobs are sequenced based on CSCOR and the regular jobs on ERD.

Finally, a combination of PSP sequencing decisions can be made. An innovative way of doing this was proposed by Abu-Suleiman et al. (2005) in the form of the *modified critical ratio* (MODCR) rule. This rule sequences the PSP based on the modified critical ratio which is equal to the critical ratio but the remaining processing time is first raised to the power of z where $0 \leq z \leq 1$. Which means that a z of 0 results in EDD while a z of 1 results in CR. All other values of z result in new unique combinations of the two rules. This method has also been used to combine other rules (e.g. (Hamidi, 2016; Kumar et al., 2017)).

3.1.2 Order Release rules an overview

The most strait forward order release rule is *Immediate release* (IMM). This rule releases every job as soon as it is available and ignoring all information on the job and the status of the shop (Ragatz and Mabert, 1988). Another rule is *Backward infinite loading* (BIL), which does not consider the shop status, however, it does consider the job's information. BIL releases a job based on the due date corrected for a fixed allowance per operation (Moreira and Alves, 2009). Ragatz and Mabert (1988) introduced a modified version of BIL which was expended on by Philipoom et al. (1993) the *Modified infinite loading* (MIL). MIL differs from BIL in that it uses a modified allowance per operation. MIL does take shop staus into account, yet it does not consider shop capacity. A rule which includes shop capacity is the *Maximum number of jobs* (MNJ). MNJ releases jobs until either all jobs are released or a predetermined maximum number of jobs are on the shop floor (Melnyk et al., 1992).

Moreira et al. (2006) introduced *Planned input-output control rule* (PIOC) to control for both the input in the number of jobs on the floor and the output delivery performance. PIOC combines aspects of BIL and MNJ as it releases a job either when it is its latest release date (as with BIL) or when the number of jobs on the shop floor falls below a predetermined maximum (as with MNJ). PIOC is very similar to *Due date and load-based release* (DLR) which has been introduced by Sabuncuoglu and Karapinar (2000). The key difference between the two methods is that PIOC uses a continues timing convention whereas DLR uses a discrete timing convention. Where PIOC and DLR aim to control both capacity and lateness *Finite forward loading* (FFL) is a rule which aims to control both the capacity and job earliness. Jobs are first scheduled in the first shift for which it would not exceed the machine's capacity and if this shift would not result in too much earliness the job is released Ahmed and Fisher (1992); Bobrowski (1989).

To better account for the different times at which released jobs arrive at different machines *Backward finite loading* (BFL), was introduced. BFL divides the planning horizon in epochs and releases jobs such that the expected capacity of a machine in each of these epochs does not exceed a predetermined maximum (Ragatz and Mabert, 1988). Similarly to BFL *Path based bottleneck* (PBB) aims to restrict the load on each machine. PBB first sorts the jobs in the pre-shop pool and evaluates the release of each job based on the current shop state and the job's characteristics. A job is released if the release of that job would not exceed the capacity of a machine on which the job has to be operated (along its path) Philipoom et al. (1993).

3.1.3 Order Release rules a classification

Based on the framework proposed by (Bergamaschi et al., 1997), OR rules can be classified based on eight distinct attributes. An overview of these attributes and possible values is given in Table 3.1 and a classification of the rules discussed in the previous section is given in Appendix A.3. The remainder of this subsection discusses these attributes. The first attribute is the *Order release mechanism* used, which can be either *load limited* or *time phased.* Load limited OR releases jobs form the PSP based on their impact on the shop and the current shop workload. Contrary, time phased OR releases jobs at computed release times regardless of the current shop workload. The second attribute by which OR is typically classified is the timing convention. OR rules with a *continuous timing convention* can release jobs at any time during which the system is active. *Discrete timing convention*, on the other hand, allows only for releases at periodic intervals. Often *load limited* rules are using a *discrete timing convention* whereas *time-phased* approaches use a *continuous timing convention.*

Workload measure is the third classification attribute for OR. Since OR usually aims to balance the workload in a shop, the exact measure of workload is of importance. Two main ways to measure workload identified in the literature are: number of jobs and work quantity. The number of jobs measure simply uses the number of jobs, while work quantity uses the expected processing times as an input. Besides the definition of the *workload measure* it is also relevant to define the *aggregation of the workload measure.* Bergamaschi et al. (1997) define three different aggregation levels. At one extreme the *total shop load*, which does not take the load distribution within the shop into account. At the other extreme the *load by each work-centre* could be used. In between these two extremes is the *bottleneck load* aggregation level. Another attribute concerns the decision on calculations of the expected workload on each work-centre. Bechte (1988) identified three types of load effecing the total workload on each work-centre: load on hand, load in transit, and released load. How OR deals with these contributes over time is the fifth classification attribute. In case no indication of load distribution over time, an OR is classified as *atemporal.* On the other hand, when the load is distributed over time buckets (e.g. shifts) the OR is classified as using *time bucketing*. Lastly, there is the *probabilistic* approach which calculates probabilities of load in transit and released load to arrive at a particular workstation for a given time. The final workload related classification attribute is *workload control* which defines what kind of approach the OR uses to control the workload. The most common approach is using an *upper workload bound* to control the WIP inventory level. Also, a *lower workload bound* could be used. In this case, the OR aims to prevent starvation of work-centres by ensuring a predefined minimum of work to be present at each station. More common is the usage of both an *upper and lower bound* to prevent starvation while still limiting the WIP inventory level. Finally, there are OR which aim to balance the overloading of one workstation with the under-loading of another (or vice-versa). When this is the case the approach is called *workload balancing*.

Furthermore OR can be classified based on how which they deal with *capacity planning*. The classical manner in which OR react to changes is by adjusting input parameters of the OR at hand based on a feedback loop. These OR are classified as *passive*. Contrary, are *active* OR, which adjust the machine capacity while the system is active. The final attribute by which OR are classified is the *schedule visibility*. This attribute considers the planning horizon taking into account by an OR, which can be either *limited* (e.g. only the closest planning period) or *extended*. In case of extended schedule visibility, the OR tolerates local bad performance if it leads to a good global state.

Table 3.1: Classification framework for Order Release adapted from Bergamaschi et al. (1997)

Attribute	Options
Order release mechanism	Load limited Time phased
Timing convention	Continuous Discrete
Workload measure	Number of jobs Work quantity
Aggregation of workload measure	Total shop load Bottleneck load Load by each work-centre
Workload accounting overtime	Atemporal Time bucketing Probabilistic
Workload control	Upper bound only Lower bound only Upper and lower bounds Workload balancing
Capacity planning	Active Passive
Schedule visibility	Limited Extended

3.2 Shop Floor Control

Similarly to the pre-shop pool sequence decision, at each machine a dispatching decision has to be made. Meaning it has to be decided in what sequence jobs in the machine's queue are processed. These sequence rules can vary greatly in complexity and information on job and shop characteristics used. There are simple rules which do not consider job nor shop characteristics and rules that do. The remainder of this section will discuss the different types of dispatching rules.

3.2.1 Simple dispatching rules

The simplest dispatching rules are rules which do not take job characteristics nor machine and shop status into account. *First come first serve* (FCF) rule gives priority to the job that arrived at the machine first (Aggarwal et al., 1973; Bulkin et al., 1966; Baker and Dzielinski, 1960). A modified version *earliest arrival date* (EAD) gives priority to the job that arrived at the shop first (Hollier, 1968; Conway, 1964; Conway et al., 2003). Contrary to FCF and EAD, *Last in first out* (LIF) and *Latest arrival date* (LAD) give priority to the jobs arriving latest at the machine and the shop respectively (Panwalkar and Iskander, 1977).

3.2.2 Job based dispatching rules

There are also rules which take job characteristics, for example due dates, into account. The simplest due date based rule is *Earliest Due Date* (EDD) (Brown, 1968; Conway et al., 2003; Eilon and Cotterjll, 1968; Jones, 1973), which gives priority to the job whose due date is earliest. A more sophisticated version of this the *Operational due-date* (ODD) gives priority to the job with the earliest operational due date (Conway, 1964; Conway et al., 2003). Meaning the due date for the specific machine it is queuing for at the moment. ODD can be further modified into the *Modified operational due date* (MOD). The modified operational due date of a job is the maximum of the jobs operational due date and the earliest possible completion time of the operation (Baker and Kanet, 1983). MOD gives priority to the job with the smallest modified operational due date. In addition to only considering the due date of a job, the current time could also be taken into account, for example, *Maximum job tardiness* (MJT). MJT gives priority to the job with the highest current tardiness.

Other rules which take jobs characteristics into account are based on the processing times of jobs without considering the due dates of the jobs. The two most basic versions of these types of rules are the *Shortest processing time* (SPT) and the *Longest processing time* (LPT) (Fryer, 1973). Which, as the names suggest, give priority to the jobs with the shortest and longest processing times at the current machine respectively. *Truncated shortest processing time* (TSPT) is an extended version of SPT, which uses SPT for jobs with a processing time below a threshold and FCF otherwise (Oral and Malouin, 1973).

An alternative would be to look at the total work remaining for each job, for example, the *Maximum work remaining* (MWKR) and *Least work remaining* (LWKR). MWKR and LWKR look at the total processing time remaining for each job and give priority to maximum and minimum processing time remaining respectively (Cigolini et al., 1998). Instead of determining the amount of work remaining best on the expected processing times the amount of work remaining could be expressed in the number of operations. *least number of operations remaining* (LOR) and *highest number of operations remaining* (HOR) are two rules which incorporate this method (Baker and Dzielinski, 1960; Conway et al., 2003; Panwalkar and Iskander, 1977).

Moreover, job characteristic based rules exist which combine due date and processing time information. Two main approaches here are slack based rules and critical ratio based rules. A job's slack is the time until a job is due minus the job's remaining processing time. Whereas a job's critical ratio is equal to time until a job is due divided by the job's remaining processing time. The simplest slack based rule is *Least slack time* (LST) which gives priority to the job with the least slack (Conway et al., 2003; Panwalkar and Iskander, 1977). Two modifications on this rule are *Least operational slack* (LOS) and *Slack per remaining operation* (SPO) (Aggarwal et al., 1973; Brown, 1968; Bulkin et al., 1966). LOS considers the current machine and gives priority to the job which has the least operational slack, where operational slack it the time until the operational due date at the current machine minus the processing time at the current machine. On the other hand, SPO considers all remaining machine and gives priority to the job with the lowest slack to operations remaining ratio. Finally, there are critical ratio based dispatching rules *Lowest critical ratio* (CR) and *Operational critical ratio* (OCR). CR gives priority to the job with the lowest critical ratio whereas OCR prioritizes jobs with low operational critical ratio (Berry et al., 1984; Panwalkar and Iskander, 1977; Baker, 1984).

3.2.3 Shop based dispatching rules

Shop based dispatching rules, dispatch jobs based on the status of the shop instead of the characteristics of the jobs. This kind of dispatching rules aims to reduce machine idleness Conway et al. (2003). *Number in next queue* (NINQ) attempts to do this by giving priority to the jobs for which the queue of the machine of the next operation contains the least jobs (Hollier, 1968). A slightly different approach is the *Least work in next queue* (LWINQ) (Panwalkar and Iskander, 1977). Similarly to NINQ, it looks to the next queue however, the priority is based on the sum of processing times in the queue of the next machine instead of the number of jobs. These methods can be extended by taking the considering the queues of the machines further in the routing as well and correcting for the number of operations until a job reaches that queue. Finally, *Look Ahead Job Demanding* (LAJD) uses a dispatching rule for each machine but overwrites this rule if a machine runs the risk of becoming idle (Holthaus and Ziegler, 1997).

3.3 Conclusion

In this chapter, a wide variety of job shop control methods has been discussed. First, methods for pre-shop pool sequencing have been discussed, then an overview of order release mechanisms, and finally possible machine dispatching rules were explored. As we have seen there is a large resemblance in pre-shop pool sequencing and machine dispatching rules. These methods can separately or together improve the control and performance of job shops. To investigate whether job shop control could be beneficial for *Parts* the next Part of this thesis will introduce a simulation model.

Part II
Model development

In this Part, we introduce a conceptual and a computerized model representing the production environment at *Parts*. In doing so, we answer our first research question: *how can a complex high-mix-low-volume job shop manufacturing environment be simulated?* The model introduced is the result of multiple iterations of the model building process proposed by Sargent (1981). An overview of this model is given in Figure 3.1.

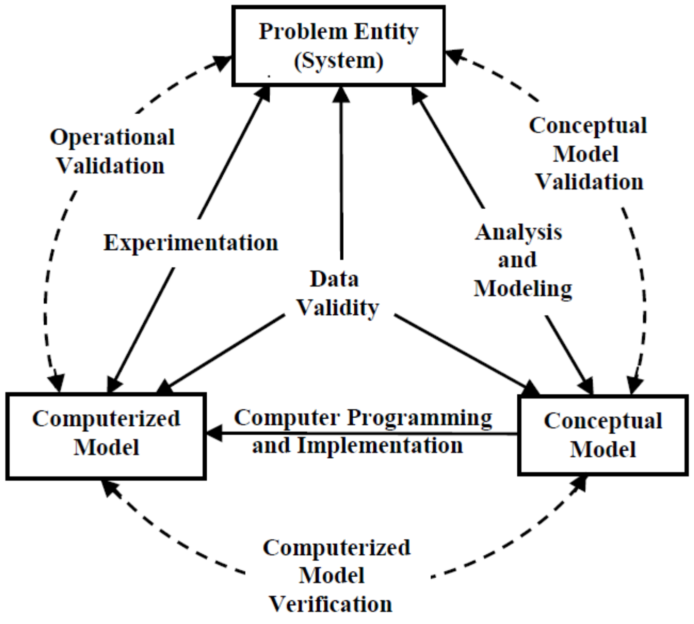

Figure 3.1: Simplified Model Building Process adopted from Sargent (1981)

Part II is organised as follows. First, the development of the conceptual model is discussed in Chapter 4. In this chapter, the structure of the model, the underlying assumptions and validity of the model are discussed. In Chapter 5 it is discussed how the conceptual model is translated into a computerized model. Chapter 5 starts with the introduction of the simulation objects. Next, the simulation is explained and the underlying distributions discussed. Thereafter, the role of sequence- and release rules is explained. Chapter 6 discusses the verification of the computerized model to the conceptual model and the operational validation of the problem entity. The verification is among others done by comparison with known mathematical models and known input output relations. The operational validation includes a reflection on the iterations of the model building process, including an overview of how the model has developed between main iterations and a reflection on the unsatisfactory earlier models.

We can roughly link the upcoming chapters to the model building process in the following manner. Chapter 4 corresponds to the Analysis and Modeling step, Chapter 5 corresponds to the computer programming and implementation step, and Chapter 6 to the Conceptual Model Validation, Computerized Model Verification, and Operational Validation steps. The Experimentation step is done in Part III. All steps are dependent on Data Validity, for a discussion on this we refer to Appendix D.

4 Conceptual Model

In this chapter, the conceptual model of *Parts* is discussed. First, in Section 4.1 the model's structure is introduced. Next, in Section 4.2 the assumptions underlying the model are discussed. This chapter ends with a discussion of the conceptual validity of the model in Section 4.3

4.1 Structure

Consider the flow of a job through a job shop as depicted in Figure 4.1. Jobs consist of several identical items which have to go through a pre-determined sequence of operations. These operations can be either internal on a machine or external and a sequence of operations is called a routing. The model starts with jobs arriving at *Parts*. These jobs are stored in the PSP, where they wait to be released. Jobs from the PSP are released to the shop at moments determined by a release rule and in a sequence determined by a sequence rule. When a job is released it either goes to an external operation or it starts queuing for an internal operation.

External operations are assumed to have infinite capacity. Internal operations, on the other hand, can only work on one job at a time. Whenever a machine becomes available for an operation it starts working on a job in its queue. The sequence in which the machine processes jobs is determined by a sequence rule. It is assumed production can not be preempted (i.e. once a machine starts processing a certain job, it will finish that job before starting on another).

Operations are subjected to yield loss, i.e. not every operation is successful on every item. If an operation fails on one or more items of a job, all defect items are converted into a new rework job and will follow an adjusted routing, while the other items continue the original routing. The adjustments made to a routing are determined by a Rework Distribution.

After a job finished an operation, it is checked whether there are any operations left to be completed in the jobs routing. If this is the case it starts queuing for the next internal operation or goes for an external operation. However, if the job does not need to complete any other operations it departs the shop, which is the last step a job can go through. Moreover, after each operation, there is a risk that a machine breaks down. In this case, it will not start on the next job until it is repaired. To conclude, the rate at which the machines process jobs can be temporarily increased if the amount of arriving work or the machine's queue pass a threshold. We call this mechanism the overtime mechanism and it is used to model all additional human interventions.

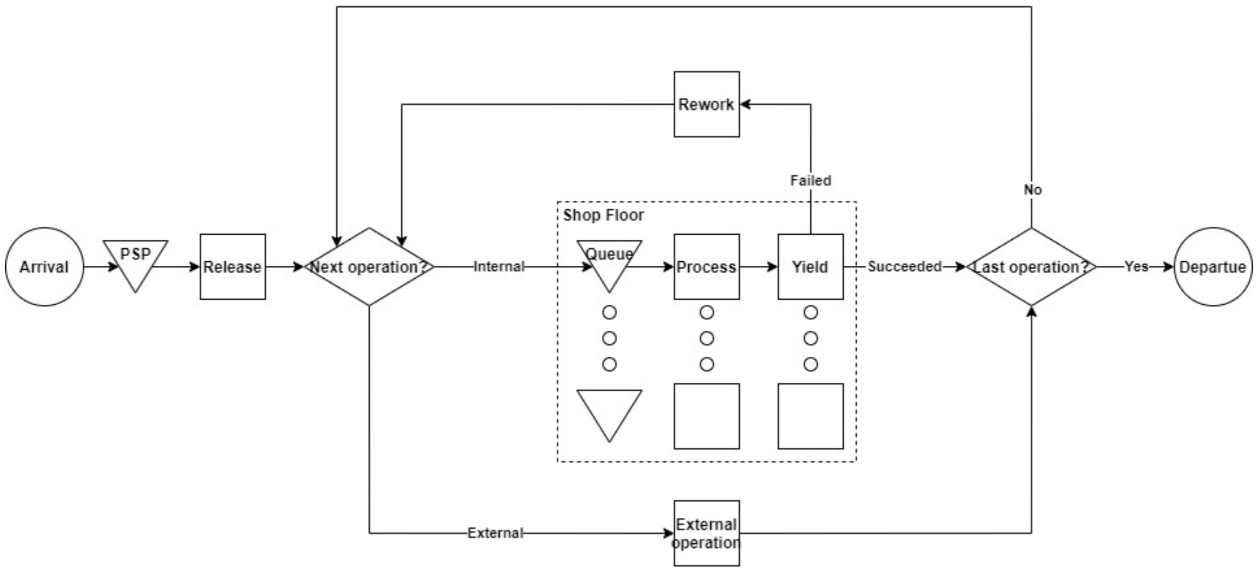

Figure 4.1: Conceptual Job Shop Model

4.2 Assumptions

The conceptual model as discussed in Section 4.1 is based on several assumptions. These assumptions are stated below and each of these assumptions is discussed briefly. For a complete overview of assumptions and a systematic representation of the model see Appendix B.

It is assumed that material is always available thus if a machine is available it can always start processing the jobs in it's queue. Also, it is assumed that all jobs that arrive at the job shop have fixed routings. Meaning it is known beforehand which machines have to process the job in which sequence. With the exception of a failed operation leading to rework, potentially imposing the routing of the rework job to additional operations. Furthermore, it is assumed that each job can only be processed on one machine, internal or external, at a time and each internal machine can only process one job at a time. Moreover, external operations are assumed to have infinite capacity. Additionally, it is assumed that setup times are not sequence-dependent. This means that the setup time of each job is independent of the job that came before it. The model also assumes the shop is always open, i.e. if there is work for machines and they are not down they run constantly. Differences between the working hours of machines is corrected by adjusting the processing rates based on these differences. Operations are never preempted. So once an operation starts it cannot be stopped until it is complete. Furthermore, items arriving at the shop are never cancelled. As a result, all jobs stay in the system until they are finished. Finally, we assume breakdowns only occur after a job is processed. Therefore, again, when an operation starts it will finish.

4.3 Validation

The combination of the structure and assumptions, introduced in Sections 4.1 and 4.2 respectively, results in the full conceptual model. The basis of the conceptual model, jobs arriving with a predetermined routing through a number of machines each with its own queue, is similar to other job shop simulation models in the literature (see Ramasesh (1990) for a list of examples). Also, previous research into the effectiveness of workload control uses the same method for modelling the PSP and order release as done in this model (Weeks, 1974; Bertrand and Van Ooijen, 2002; Thürer et al., 2017). We did not find any previous job shop simulation studies including all three disturbances, yield loss, machine breakdowns, and external operations, in the same model. However, machine breakdowns (Holthaus, 1999) and external operations (Brown et al., 2015) have been included in separate job shop simulation models before.

Moreover, the addition of all three the disturbances is the result of multiple iterations of model building, testing, and validating. In each iteration a conceptual model was built based on weekly interviews with team leaders of supply chain and logistics. Afterwards, a computerized model was made and this model has been tested by comparing the model outcomes to the actual situation at *Parts*. When the results of the model and the actual results did not sufficiently align, a revised conceptual and computerized model were made. This iterative process was repeated until the conceptual model presented in this chapter resulted in a computerized model yielding satisfactory results. Which aspects of the model changed throughout this process is discussed in depth in Chapter 6.

Since the modelling methods used are common in literature and because the structure and assumptions of the model are validated by the domain knowledge experts through interviews, we conclude that this conceptual model is a valid representation of the job shop production process at *Parts*.

5 Computerized Model

In the Chapter 4, the conceptual model has been introduced. This chapter explains how this conceptual model is translated into a computerized model. The computerized model is made in Python using the discrete event simulation library Simpy (Lünsdorf and Scherfke, 2013). In this simulation objects and events interact with each other to simulate the behaviour described in the conceptual model. Moreover, events use distributions and rules to determine their behaviour. Note that the model presented in this chapter is the result of several iterations of the model building process shown in Figure 3.1.

This chapter is structured as follows, first, in Section 5.1, the objects within the simulation are explained. Then, Section 5.2 describes how time flows in the simulation and how events can interact with these objects. Next, Section 5.3 gives a detailed overview of each event in the simulation. Sections 5.4 and 5.5 explain the structure of distributions and rules respectively. Thereafter, Section 5.6 describes the collection of key performance indicators (KPIs) throughout the simulation. This chapter ends with a conclusion in Section 5.7.

5.1 Objects

The simulation uses three types of objects to model the system, jobs (Subsection 5.1.1), machines (Subsection 5.1.2), and the pre-shop pool (Subsection 5.1.3). Each of these objects is characterised by several attributes and can have one or more states. In this section, an overview of these objects, their attributes, and states is given.

5.1.1 Jobs

Each job j is characterised by the following attributes: size s_j, arrival time a_j, due time d_j, priority p_j, and original routing $\mathcal{O}_j^{OG}$. In the simulation a routing is an ordered list of machines on which the job has to be processed. Also, the remaining routing $\mathcal{O}_j^{RM}$ (i.e. the routing steps still to be completed) and the finished routing $\mathcal{O}_j^{DN}$ (i.e. the routing steps already completed) are stored. The variable characterising jobs is a Boolean variable (x_j) describing whether the items in the job have been repaired or not. Table 5.1 gives an overview of these attributes. Moreover, the model contains four distinct states for jobs, listed in Table 5.2, indicating the status of the job. These states are: in the pre-shop pool (*psp*), in the queue for a machine (*que*), being processed (*pro*), and left the shop (*dep*).

Table 5.1: Overview of job attributes

Attribute	Notation	Description
Size	s_j	the number of items in the job
Priority	p_j	the priority of the job
Arrival time	a_j	time the job arrived at the shop
Due time	d_j	time the job is due
Original routing	$\mathcal{O}_j^{OG}$	original routing of the job
Remaining routing	$\mathcal{O}_j^{RM}$	remaining routing of the job
Finished routing	$\mathcal{O}_j^{DN}$	finished routing of the job
Rework	x_j	items of the job are repaired or not

Table 5.2: Overview of job states

Status	Notation
In the pre-shop pool	*psp*
In a machine queue	*qeu*
In process	*pro*
Departed	*dep*

5.1.2 Machines

Each machine m is characterized by the following attributes, also listed in Table 5.3 mean production time μ_m, production time standard deviation σ_m, yield y_m, down frequency ν_m, downtime λ_m, and queue q_m. The queue of a machine consists of jobs waiting to be processed on that specific machine. Similarly to jobs, the model assigns distinct states to machines indicating its current status. These three states, also listed in Table 5.4, are: idle (*idl*), processing (*pro*), and down (*dwn*).

Table 5.3: Overview of machine attributes

Attribute	Notation	Description
Process time mean	μ_m	the mean processing time of a job on this machine
Process time variation	σ_m	the standard deviation of processing time of a job on this machine
Yield	y_m	the probability an operation on this machine is successful
Down frequency	ν_m	the probability this machine goes down
Down time	λ_m	the mean time this machine is down
Queue	q_m	the jobs waiting for this machine

Table 5.4: Overview of machine states

Status	Notation
Idle	*idl*
Processing	*pro*
Down	*dwn*

5.1.3 Pre-shop pool

The pre-shop pool is an object as it has only one attribute queue q_0. This queue consists of all jobs waiting to be released to the shop floor. If the queue is empty the status of the pre-shop pool is also empty (emp) and occupied (ocp) otherwise. Table 5.5 gives an overview of these states.

Status	Notation
Empty	*emp*
Occupied	*ocp*

Table 5.5: Overview of pre-shop pool states

5.2 Simulation

Section 5.1 gave an overview of the objects used in the simulation. In this section we will introduce how the simulation is set-up, time is progressed, and events are called. Algorithm 5.1 gives a systematic overview of how this works while the remainder of this section gives a textual description.

The start of the simulation is the initialisation of the environment. For this initialisation, the simulation time is set and objects are created. First, the simulation time is set to zero ($t_{now} = 0$). After which all machines and the pre-shop pool (psp) are created. Note that Jobs are not created beforehand since they enter and depart the shop throughout the simulation. Instead, the first order arrival event is scheduled at $t = 0$, which creates the first job and schedules the next order arrival. How events schedule other events is explained below and the exact dynamics of order arrival are explained in Section 5.3.

After all objects are created the simulation loop starts. This loop continues until the simulation time reaches a specified end time (t_{end}). During this simulation loop, time advances using the next-event time-advance approach (Law et al., 2000). This approach utilises a set of scheduled events each consisting of parts; the interactions with the simulation objects; and the events it schedules. This set of scheduled events is used as follows. The first scheduled event is taken from the set, then the simulation time advances to the time at which this event is scheduled. Then interactions with the simulation objects are handled and new events are scheduled. When the simulation end time is reached KPIs are calculated. How the KPIs are calculated is explained in Section 5.6.

Algorithm 5.1 Simulation

```
 1: $t_{now} \leftarrow 0$
 2: for 1 to number of machines do
 3:     create machine
 4: end for
 5: create psp
 6: schedule Order Arrival at $t = 0$
 7: while $t_{now} \leq t_{end}$ do
 8:     next event ← first event from scheduled events
 9:     $t_{now} \leftarrow$ time of next event
10:     if type of next event is <type> then
11:         do change objects based on <type>                // (see: Section 5.3)
12:         schedule events based on <type>
13:     end if
14: end while
15: calculate KPIs                                           // (see: Section 5.6)
```

Note that the simulation does not distinguish between work hours and closed hours or workdays and holidays, but runs as if every machine in the shop runs constantly. This is done to streamline the simulation. One time unit in the simulation corresponds to one working hour of an eight-hour workday. To facilitate this all parameters have been converted to represent this continuous working system. How the parameters are adjusted is discussed upon introduction of each parameter (mostly in Section 5.4).

5.3 Events

Section 5.2 provided an overview of the global behaviour of the simulation and the role of events. This section describes each event in detail. Events use distributions and rules as input. These distributions and rules are explained in Sections 5.4 and 5.5 respectively. An overview of events, their interactions with the objects, and events they schedule is given in Appendix C.

5.3.1 Order Arrival

The event Order Arrival is used to create jobs and model their arrival. A systematic representation of the event is given in Algorithm 5.2. It works as follows. First, the job's routing, size, due date, and priority are sampled from a routing-, size-, due date- and priority distribution respectively. These distributions are explained in Subsections 5.4.2, 5.4.3, 5.4.4, and 5.4.5. Afterwards a new job with the sampled values as parameters is created. Next, the job is added to the pre-shop pool and its state is set to psp. Then Order Release is scheduled at the current simulation time. Finally, the next Order Arrival has to be scheduled. This is done by sampling an inter-arrival time $t_{arrival}$ from an arrival distribution (Subsection 5.4.1) and scheduling Order Arrival at the current simulation time plus the interarrival time.

5.3.2 Order Release

The event Order Release is used to release jobs from the pre-shop pool following a Sequence- and a Release Rule. Sequence- and Release Rules are explained in Subsection 5.5.1. A systematic representation of the Order Release is given in Algorithm 5.3. The Order Release event first checks whether the criteria for order release are met. If this is the case the first job is taken from the pre-shop pool based on a Sequence Rule. Then a Next Operation event is scheduled for this job at the current simulation time. Finally, a new Order Release event is scheduled at the current simulation time.

Algorithm 5.2 Order Arrival

```
 1: sample $r$ from Routing Distribution                    // (see: Section 5.4.4)
 2: sample $s$ from Size Distribution                       // (see: Section 5.4.2)
 3: sample $d$ from Due Date Distribution                   // (see: Section 5.4.5)
 4: sample $p$ from Priority Distribution                   // (see: Section 5.4.3)
 5: job is new job with
            original routing $\leftarrow r$
            remaining routing $\leftarrow r$
            size $\leftarrow s$
            due date $\leftarrow d$
            priority $\leftarrow p$
 6: add job to psp
 7: set state of job to psp
 8: sample $t_{arrival}$ from Arrival Distribution          // (see: Section 5.4.1)
 9: schedule Order Arrival at $t_{now} + t_{arrival}$
10: schedule Order Release at $t_{now}$
```

Algorithm 5.3 Order Release

```
1: if Release Criteria is True then                         // (see: Section 5.5.2)
2:     $j \leftarrow$ job from psp based on Sequence Rule   // (see: Section 5.5.1)
3:     schedule Next Operation at $t_{now}$ with $j$
4:     schedule Order Release at $t_{now}$
5: end if
```

5.3.3 Next Operation

The event Next Operation is used to determine what the next step for jobs will be. A systematic representation of the event is given in Algorithm 5.4. Next Operation is always scheduled for a specific job. First it checks if there are any operations left in the routing of the job. If this is the case it is either an external or internal operation. In case of an external operation, the event External is scheduled for the job at the current simulation time. In case of an internal operation, the machine for that internal operation is called. Next, the job is added to the queue of that machine and the job's state is set to *que*. Thereafter, Check Overtime Policy is scheduled for the machine at the current simulation time. If, there are no operations left in the job's routing, the job leaves the shop and its state is set to *dep*.

Algorithm 5.4 Next Operation

```
Require: job
 1: if remaining routing of job is not empty then
 2:     next operation $\leftarrow$ first operation from remaining routing
 3:     if next operation is external then
 4:         schedule External at $t_{now}$ with job
 5:     else
 6:         next machine $\leftarrow$ machine of next operation
 7:         add job to queue of next machine
 8:         set state of job to que
 9:         schedule Check Overtime Policy at $t_{now}$ with next machine
10:     end if
11: else
12:     set state of job to dep
13: end if
```

5.3.4 Check Overtime Policy

The event Check Overtime Policy is used to determine whether it is necessary to either stop or start running overtime. Check Overtime Policy is always scheduled for a machine. The operation Check Overtime Policy operates similar to thermostats and a systematic overview is given in Algorithm 5.5. First, it is checked whether the machine is currently working overtime. If so, it is checked whether both the queue at the machine and the work planned to arrive at the machine are both below the threshold for stopping with running overtime. If this is the case the policy of the machine is switched to not running overtime. On the other hand, in case the current policy of the machine is to not run overtime it is checked whether either the queue at the machine or the work planned to arrive at the machine is above the threshold to start running overtime. If this is true the machine's policy is changed to running overtime. Note that it is only necessary to exceed one of the upper limits to start working overtime while both the lower limits need to be met to stop. Afterwards, the event Start Operation is scheduled for the machine at the current simulation time.

Algorithm 5.5 Check Overtime Policy

```
Require: machine
 1: planned ← planned work of machine
 2: queue ← queue of machine
 3: maxQueue ← upper queue limit of machine
 4: minQueue ← lower queue limit of machine
 5: maxPlanned ← upper planned work limit of machine
 6: minPlanned ← lower planned work limit of machine
 7: if policy of machine is no overtime then
 8:     if queue ≥ maxQueue or plannedwork ≥ maxPlanned then
 9:         set policy of machine to overtime
10:     end if
11: else
12:     if queue ≥ minQueue and plannedwork ≤ minPlanned then
13:         set policy of machine to no overtime
14:     end if
15: end if
16: schedule Start Operation on $t_{now}$
```

5.3.5 Start Operation

The event Start Operation is used to determine if a machine can start processing a job and, if this is the case, start the processing. A systematic representation of this event is given in Algorithm 5.6. Naturally, Start Operation is always scheduled for a specific machine. First, we verify there are jobs in the queue for the machine and whether the machine is available for processing (i.e. its state is *idl*). In case these criteria are met, the first job is taken from the queue based on a Sequence Rule. Sequence Rules are explained in Subsection 5.5.1. Next, the job's state is set to *pro*. Subsequently, the state of the machine is also set to *pro*. Finally, a processing time $t_{process}$ is sampled from a process distribution (Subsection 5.4.6) and the event Finish Operation is scheduled for the job and the machine at the current time plus the processing time.

Algorithm 5.6 Start Operation

```
Require: machine
 1: queue ← queue of machine
 2: if status of machine is idl then
 3:     if queue is not empty then
 4:         job ← job from queue based on Sequence Rule          // (see: Section 5.5.1)
 5:         set state of job to pro
 6:         set state of machine to pro
 7:         sample $t_{process}$ from process distribution       // (see: Section 5.4.6)
 8:         schedule Finish Operation on $t_{now} + t_{process}$ with job, machine
 9:     end if
10: end if
```

5.3.6 External

The event External is used to model external operations on jobs. A systematic representation of the event is given in Algorithm 5.7. External is scheduled for a specific job and works as follows. First, the state of the job is set to *pro*. Next, the external processing time $t_{external}$ is sampled from an external distribution. Finally, the event Next Operation is scheduled for the job at the current simulation time plus the external processing time.

Algorithm 5.7 External

```
Require: job
1: set status of job to pro
2: sample $t_{external}$ from process distribution               // (see: Section 5.4.6)
3: schedule Next Operation on $t_{now} + t_{external}$ with job
```

5.3.7 End Processing

The event End Processing is used to model the end of an operation on a job. This includes potential yield issues and machine failures. A systematic representation of the event is given in Algorithm 5.8. This event handles the machine object first and thereafter the job object.

For the machine, first a downtime t_{down} is sampled from the Down Distribution. In case this downtime is larger than zero the machine goes down. This is modelled by setting the machine's state to *dwn* and scheduling the Repaired event for the machine at the current simulation time plus the downtime. In case the downtime is zero, the machine state is set to *idl* and Check Overtime Policy is scheduled for this machine at the current simulation time.

Handling the job starts with checking the number of items on which the operation was successful and on which the operation failed. If there are any failed operations a rework job is created including all items on which the operation failed and the remaining items are clustered in a regular job. Next, the event Rework is scheduled for the rework job at the current simulation time. If the operation was successful on any of the items, Next Operation is scheduled for the job with these items (excluding the items now in the rework job) at the current simulation time. If there are no successful items, the original job is deleted.

Algorithm 5.8 End Processing

```
Require: $job$, $machine$
 1: sample $t_{down}$ from $down\ distribution$            // (see: Section 5.4.7)
 2: if  $t_{down} > 0$ then
 3:     set $state$ of $machine$ to $dwn$
 4:     schedule $Repaired$ at $t_{now} + t_{down}$ with $machine$
 5: else
 6:     set $state$ of $machine$ to $idl$
 7:     schedule $Check\ Overtime\ Policy$ at $t_{now}$ with $machine$
 8: end if

 9: $size$ is $size$ of $job$
10: sample $failed$ from $Yield\ Distribution$            // (see: Section 5.4.8)
11: $success$ is $size - failed$
12: if $failed > 0$ then
13:     $rework\ job \leftarrow$ copy $job$
14:     $size$ of $rework\ job \leftarrow failed$
15:     Schedule $Rework$ at $t_{now}$ with $rework\ job$
16: end if
17: if $success > 0$ then
18:     $size$ of $job \leftarrow success$
19:     Schedule $Next\ Operation$ at $t_{now}$ with $job$
20: else
21:     delete $job$
22: end if
```

5.3.8 Repaired

The event Repaired is used to model the end of a machine's down period. A systematic representation of the event is given in Algorithm 5.9. This event handles a specific machine. First it changes the state of that machine to idl. Finally, it schedules Check Overtime Policy for that machine at the current simulation time.

Algorithm 5.9 Repaired

```
Require: $job$
 1: set $status$ of $machine$ to $idl$
 2: schedule $Check\ Overtime\ Policy$ at $t_{now}$ with $machine$
```

5.3.9 Rework

The event Rework models how the routing of rework jobs are adjusted. A systematic representation of the event is given in Algorithm 5.10. This event handles a job. First, it checks if the items already have been repaired. If this is not the case, the last routing step has to be done over. This is done by copying this step from the finished routing of the job to the remaining routing of the job. However, if the items already have been repaired the items are scrapped. In this case the remaining routing is replaced with the original routing. After the routing of the job has been changed the event Next Operation is scheduled for the job at the current simulation time.

Algorithm 5.10 Rework

```
Require: $job$
1: $\mathcal{O}^{OG} \leftarrow$ original routing of $job$
2: $\mathcal{O}^{DN} \leftarrow$ finished routing of $job$
3: $\mathcal{O}^{RM} \leftarrow$ remaining routing of $job$
4: if $repaired$ is False then
5:     $machine$ is last machine of $\mathcal{O}^{DN}$
6:     add $machine$ at start of $\mathcal{O}^{RM}$
7: else
8:     clear $\mathcal{O}^{RM}$
9:     copy $\mathcal{O}^{OG}$ to $\mathcal{O}^{RM}$
10: end if
11: schedule $Next\ Operation$ on $t_{now}$ with $job$
```

5.4 Distributions

In Section 5.3 a detailed explanation of the events in the simulation was given. The exact behaviour of these events is influenced greatly by the distributions used as their input. Nine different types of distributions are used in the simulation. One is used to model times at which jobs arrive (Arrival Distribution), four are used to generate new jobs (Size Distribution, Priority Distribution, Routing Distribution, and Due date Distribution), and four are used to model the dynamics of internal operations (Process distribution, Down distribution, Yield Distribution, and Rework Distribution).

These distributions are based on a data set of historic data obtained, and interviews conducted at *Parts*. The historic data contains three years of job and machine information, including arrival times, routings, priorities, due dates, waiting times, and process times. For details on the process of data collection, a discussion on data quality, and the steps of data cleaning see Appendix D. The interviews consisted of weekly interviews with team leaders of the supply chain engineering and logistics departments and several interviews with employees from the maintenance- and production department. The remainder of this section describes the distributions used in the simulation. For a complete overview of simulation parameters see Appendix E.

5.4.1 Arrival Distribution

The Arrival Distribution determines how much time there is between the arrival of two separate jobs (i.e. the interarrival time). It is common for jobs in job shop simulations to arrive according to a Poisson process (Weeks, 1974; Bertrand and Van Ooijen, 2002; Thürer et al., 2017). Therefore the interarrival time is modelled as an exponential distribution with rate $\lambda_{arrival}$. By taking the mean number of arrivals in a year from the historic data and correcting for the work hours in a year the arrival parameter was estimated to be $\lambda = 2.62$. This means that on average every simulation time unit 2.62 jobs arrive at the shop.

5.4.2 Size Distribution

Size Distributions are used to model the number of items in a job. The reason jobs are modelled to have different sizes is so they can be split different times into rework and regular jobs in case of yield loss. This effect is created in a sufficient amount by modelling the size of a job by a discrete uniform distribution on the interval $[1, 5]$.

5.4.3 Priority Distribution

Priority Distributions determine the priority of arriving jobs to the shop. Priorities are modelled by integers, where the larger the number the higher the priority of the job. The fraction of jobs belonging to each priority group in the historic data is tabulated in Table 5.6. Based on these fractions, intervals are created such that, drawing from a uniform distribution on the interval $[0, 1]$ and assigning a priority based on the interval the resulting number falls in, results in a similar distribution of priorities.

Table 5.6: Distribution of job priorities

Priority Level	Fraction of jobs	Corresponding interval
1	68.1%	(0.000, 0.681)
2	3.2%	(0.681, 0.731)
3	5.6%	(0.731, 0.769)
4	13.3%	(0.769, 0.902)
5	9.8%	(0.902, 1.000)

5.4.4 Routing Distribution

Routing Distributions determine the routing of a job. Meaning it has to return an ordered list of integers corresponding to machines. This is modelled as an empirical distribution, essentially drawing randomly from a set of predetermined routings. This empirical distribution is made by collecting three years worth of routing data and sampling with replacement from this distribution.

5.4.5 Due date Distribution

The due dates of jobs are determined by Due date Distributions. Our analysis of the historic data showed that a relationship exists between a job's expected processing time and the time until it is due. This relationship is such that a job with a longer expected processing time has on average more slack upon arrival than a job with shorter expected processing time.

To model this an empirical distribution is made of the ratio between the processing time of a job and the time until that job is due. We call this ratio a jobs due-date tightness. The due date tightness of a job is less then one if the time until it is due is less than the expected processing time, greater than one if the time until it is due is greater than the expected processing time, and one otherwise. The due date of a job is determined based on this distribution by multiplying a number drawn from this distribution by the expected processing time of that job.

5.4.6 Process Distribution

The Process Distribution is used to determine the time needed for an operation of a job on a machine and is build up from two parts. A basic part based on the historical process time and a factor to account for overtime.

The distribution used for the basic part is determined by considering four common processing times distributions: the uniform-, exponential-, normal-, and log-normal distribution (Baker and Trietsch, 2013). The historical data of process times for each machine was used to test each of these process times distributions using a Chi-squared test. For most machines, a log-normal distribution turned out to be the best fit. Therefore, we decided to use a log-normal distribution for all machines. Based on the historical data we estimated the mean and variation of process time at each machine.

As mentioned in Section 5.2 one unit of simulation time corresponds to one work hour in an eight-hour workday. However, at *Parts* different machines work for a different number of hours per day. To account for this the processing time mean and variation of each machine is corrected. This is done in similar to Brown et al. (2015), by multiplying the processing time mean and variation of each machine by the ratio of actual machine capacity to eight hours of machine capacity. The corrected mean and variation for each machine are shown in Appendix E. In short, the basic part of the processing time is a number sampled from a log-normal distribution with mean and variations obtained from historical data corrected for machine capacity.

The number obtained from this distribution is used directly in case the machine is not running overtime. However, if the machine is running overtime the machine has additional capacity which should be accounted for. This is done by multiplying the processing time by the inverse of the increase of capacity as the result of overtime, which we call the overtime rate. The estimation of this overtime rate was made during interviews with the team leader of the supply chain engineers and is 0.909.

5.4.7 Down Distribution

The Down Distribution is used to determine two things, whether a machine goes down and how long it will be down. This is done by first drawing from a Bernoulli distribution with parameter ν determining whether the machine goes down or not. If the machine goes down the duration of the downtime is modelled by an exponential distribution with parameter λ_m. If the machine does not go down, zero is returned. Data on machine availability was scarce and therefore this distribution and its parameters are based on several interviews with employees from the maintenance- and production department. The estimated downtime parameter is $\nu = 0.1$ and a machine is expected to be down for as long as it is expected for the machine to complete one job $\lambda_m = t_{process}$.

5.4.8 Yield Distribution

Yield Distributions determine the number of failed operations. This is modelled by a binomial distribution with size parameter s_j and probability parameter y_m. s_j is equal to the size of the job for which the yield is determined. y_m is based on the average machine yield of the past three years. Similar to the down distribution, data on machine yield was scarce and therefore this distribution and its parameters are based on several interviews with employees from the maintenance- and production department. The estimation for the machine yield parameter is $y_m = 0.985$.

5.4.9 External Distribution

External Distributions determine the duration of external operations. The process of determining this distribution was the same as for the Process Time distribution. Meaning Chi-squared tests have been used to test which of the four common processing times distributions fits best to the historical data of the duration external operations. These tests showed that a log-normal distribution was the best fit. Moreover, the mean and variation of the duration of external operations was determined based on the historical data and subsequently transformed into simulation time units. As a result the duration of external operations is modelled by a log-normal distribution with parameters $\mu = 87.47$ and $\sigma = 297.96$.

5.5 Rules

The simulation uses two types of rules to model the decisions made in the system; Sequence Rules determine in what sequence to release and process jobs, and Release Rules determine when to release jobs. In this section these rules are explained and examples are given. Formal definitions used in this study of all rules are given in Chapter 7.

5.5.1 Sequence Rule

Sequence Rules are used to determine in what sequence jobs should be released or processed depending on whether they are applied to the pre-shop pool or machine queues. All sequence rules take a list of jobs as input and return the job that has the highest priority based on the specific rule. The basic method used in this thesis is the First Come First Serve rule, which sorts jobs based on their arrival time. Algorithm 5.11 shows how this works. The other sequence rules tested are formally introduced in Chapter 7.

Algorithm 5.11 First Come First Serve

```
Require: jobs
1: current job is first job in jobs
2: current arrival time is arrival time of current job
3: for new job in jobs do
4:     new arrival time is arrival time of new job
5:     if new arrival time < current arrival time then
6:         current job is new job
7:         current arrival time is new arrival time
8:     end if
9: end for
10: return current job
```

5.5.2 Release Rule

Release Rules determine whether a job can be released from the pre-shop pool onto the shop floor. Release Rules take the current shop status as input and return True in case a job can be released and False in case no job can be released. The current shop status is the collection of all objects. One example of a release rule is IMM. IMM releases every job immediately upon arrival and thus corresponds to having no pre-shop pool. A systematic representation is given in Algorithm 5.12. As can be seen this release rule returns True as long as there is a job in the pre-shop pool.

Algorithm 5.12 Immediate release

```
Require: jobs
1: if length of jobs ≥ 1 then
2:     return True
3: else
4:     return False
5: end if
```

5.6 Monitoring

Sections 5.1, 5.3, and 5.4 explained how events and objects interact in the simulation model to model *Parts*. In this section, we explain how this simulation is used to obtain KPIs on the shop performance for different settings. Monitoring the simulation to obtain the relevant KPIs is done by tracking the shop status. As mentioned before, the shop status is the collection of all objects in the system. Every time the shop status changes the time of this change and the new shop status are saved. Additionally, the start and end times of machine operations are stored. In doing so, all necessary information for calculating KPIs is stored.

5.7 Conclusion

In this chapter, the computerized model was introduced. This model consists of three types of objects, Jobs, Machines and Pre-Shop Pool. These objects are modified by the events, which schedule new events to ensure the continuation of the simulation. Additionally, distributions and rules were introduced. Distributions are used to model the stochastic behaviour of the simulation whereas rules are used to model the decisions made in the system. These objects, events, distributions and rules are used to collect KPIs via monitoring functions. The model presented in this chapter is the result of an iterative process following the model building process proposed by Sargent (1981) and introduced at the start of this Part.

6 Validation and Verification

Chapter 5 discussed the computerized model. In this chapter, the verification and validation of the computerized model are explained and the sensitivity of the model discussed. Also we provide the answer to the first sub-research question:

- **SRQ1**: How can a complex high-mix-low-volume job shop manufacturing environment be simulated?

This chapter is organised as follows. Section 6.1 discusses the verification of the model, which is based on determining whether the computerized model behaves according to the conceptual model. Next, the operational validation or the relationship between the computerized model and the actual situation at *Parts* is discussed in Section 6.2. We end this Chapter with a summary in Section 6.3.

6.1 Computerized Model Verification

The objective of computerized model verification is defined as: "assuring that the computer programming and implementation of the conceptual model are correct" (Sargent, 1981). The simulation model verification is done by three different techniques: traces, face validity, and comparison with other models. Each of the methods is discussed in the upcoming subsections.

6.1.1 Traces

Tracking and storing the behaviours of different types of specific entities in the model throughout time results in traces. These traces are used to determine whether the model's logic is correct and if it behaves as intended. Examples of traces which have been checked are the operations an job goes through and the machine queue throughout time. The traces obtained from the model all corresponded to behavior as expected from the conceptual model.

6.1.2 Face Validity

The face validity of a model can be checked by using domain expert knowledge to check whether model behaviour for certain settings is reasonable. We have tested whether the model adheres to 22 known input-output relations from queuing theory. An overview of these relations is given in Appendix E.1. The computerized model adheres to all of these relations.

6.1.3 Comparison to Other Models

By simplifying the simulation model we can compare the model with other valid (mathematical) models. Here we discuss the mathematical models used the verify the simplified computerized model. For comparison of the mathematical and simulation results see Appendix F

m/m/1 By simplifying the simulation model to a situation with a Poisson arrival process, first come first serve (FCFS) processing, and one machine with exponential processing time we end up with a m/m/1 queue (Sztrik, 2012). Let ρ be the server utilisation, μ the service rate, and λ the arrival rate. Then it is easily shown that for a m/m/1 queue the expected number of customers in the system in steady-state is:

$$\bar{N} = \frac{\rho}{1-\rho} \tag{2}$$

The expected waiting time of a customer is:

$$\bar{W} = \frac{\rho}{\mu(1-\rho)} \tag{3}$$

Finally, the expected total time a customer spends in the system is:

$$\bar{T} = \frac{1}{\mu(1-\rho)} \tag{4}$$

m/m/1 with priority Again consider a simplified version of our simulation model with an arrival rate λ_i for jobs of priority type i. Jobs are processed FCFS, however jobs with a lower priority level have non-preemptive priority (e.g. when a new job can start an job with priority i gets picked over an job with priority $j > i$, but processing of jobs is not interrupted). Let ρ_i be the fraction of the time the machine is busy on jobs with priority i. Then it is known from queuing theory that in the case of two priority levels the expected steady-state number of customers of type 1 and type 2 is equal to (Sztrik, 2012):

$$\bar{N_1} = \frac{(1+\rho_2)\rho_1}{1-\rho_1} \tag{5}$$

$$\bar{N_2} = \frac{(1-\rho_1(1-\rho_1-\rho_2))\rho_2}{(1-\rho_1)(1-\rho_1-\rho_2)} \tag{6}$$

Moreover, it is known that the expected waiting time of jobs of type 1 and type 2 is equal to:

$$\bar{W_1} = \frac{(1+\rho_2)\mu}{1-\rho_1} \tag{7}$$

$$\bar{W_2} = \frac{(1-\rho_1(1-\rho_1-\rho_2))\mu}{(1-\rho_1)(1-\rho_1-\rho_2)} \tag{8}$$

Jackson Network Consider a Jackson network with n machines each capable of processing one job at the time with service rate μ_i for machine i (for more information on Jackson networks see (Boucherie and Van Dijk, 2010)). Let jobs arrive from outside at machine i with rates λ'_i. Let the probability of an job processed on machine i going to machine j be transition rate $p_{i,j}$. When the transition rates do not add up to 1 jobs also leave the system. Arrival rates λ_i at each machine can be determined by solving the set of balance equations:

$$\lambda_i = \lambda'_i + \sum_{j=1}^{n} p_{i,j}\lambda_j \tag{9}$$

Then by treating each machine as a m/m/1 server, it can be shown that the expected number of customers in the system is

$$\bar{N} = \sum_{i=1}^{n} \frac{\rho_i}{1-\rho_i} \tag{10}$$

As a result, the expected waiting time of a customer is equal to

$$\bar{W} = \bar{N} \sum_{i=1}^{n} \frac{1}{\lambda'_i} \tag{11}$$

Yield with Scrapping Consider the situation in which an operation is successful with probability α. Each time an operation fails the semi-finished product is scrapped and the job has to start over from the beginning. Let n be the expected initial routing length of an job then the expected number of operations needed before an job is finished is equal to:

$$N = \frac{1}{\alpha} + \frac{1}{\alpha^2} + ... + \frac{1}{\alpha^n} = \frac{1-\alpha^n}{\alpha^n(1-\alpha)} \tag{12}$$

Yield with a Single Repair Step and Scrapping Consider this situation slightly adjusted from the one above. If an operation fails the semi-finished product is repaired. However, when an operation fails on a repaired semi-finished product, the product is scrapped and the job has to start over from the beginning. In this situation three things can occur: an job can finish after n operations without any failures, an job can finish after $n+1$ operations with just one failure or an job is scraped and has to start over after $1 < x < n+1$ operations. The probability an job finishes without any failures is equal to:

$$P_0 = \alpha^n \tag{13}$$

The probability an job finishes with exactly one failed operation is equal to:

$$P_1 = n\alpha^n(1-\alpha) \tag{14}$$

The probability an job being scrapped after x operations is equal to $(x-1)\alpha^{x-2}(1-\alpha)^2$ and therefore the probability an job gets scrapped:

$$P_2 = \sum_{x=2}^{n+1}(x-1)\alpha^{x-2}(1-\alpha)^2 \tag{15}$$

The expectation of additional steps a job has to go through after being scrapped are equal to the expectation of the number of steps a job has to go through at the start; N. Using equations 13, 14, and 15 then by definition the expected number of operations a job undergoes before finishing is equal to:

$$N = n\alpha^n + (n+1)n\alpha^n(1-\alpha) + \sum_{x=2}^{n+1}(x+N)(x-1)\alpha^{x-2}(1-\alpha)^2 \tag{16}$$

Which can be rewritten into:

$$N = \frac{n\alpha^n + n(n+1)\alpha^n(1-\alpha) + (1-\alpha)^2\sum_{x=1}^{n}(x+1)x\alpha^{x-1}}{1-(1-\alpha)^2\sum_{x=1}^{n}x\alpha^{x-1}} \tag{17}$$

The simplified computerized models resulted in similar values as the m/m/1, m/m/1 with priority, Jackson Networks, and yield formulas. Therefore we conclude that verification by comparison to other models provides evidence for the verification of the model. In short, traces, face validity and comparison to other models all provide evidence for the positive verification of the model, and therefore we conclude the model is positively verified.

6.1.4 Sensitivity analysis

We evaluate the models sensitivity to parameter estimations and design decisions. To test the models sensitivity to parameter estimations we slightly adjusted eight parameters up and down and evaluated the resulting change in WIP and effective throughput. For a full overview of results of the tests see Appendix G. These tests show that the WIP changes more than the throughput to adjustments in the input parameters. The model is sensitive to are the arrival rate and the processing time mean. This is explained by the fact that *Parts* operates at a high utilization and thus relative small changes in arrival rate or processing rate have a high impact on the model. The model is also sensitive to changes in the external operation time mean, which is the time it takes for a job to return after going for an external operation. This time includes potential waiting and transport time. Consequently, a one percent increase or decrease constitutes to a large absolute change in necessary time.

6.2 Operational Validation

In Section 6.1 the verification of the computerized model, which checks the relation between the conceptual and the computerized model, has been discussed. This section discusses the operational validation. When attempting to create an operationally valid computerized model two aspects are particularly important to consider. The first is the difference between long term average behavior of a system versus the typical behavior of a system. And secondly, the realisation that in modeling inherently assumptions are made and aggregation is done, this section discusses both these aspects.

6.2.1 Empirical Validation

In this subsection we discuss the empirical validity of the model by comparing the values of KPIs obtained though simulation with the three year averages obtained from *Parts*. Moreover, we discuss the the model building process more in depth and reflect on the main iterations made in this process.

The KPIs we verify the empirical validity of the model on are: delivery performance, production time, and work in progress. Based on historical data collected we calculated the three year averages for these KPIs. The delivery performance was 14.1%, production time was to 113.26 hours, and work in progress was 1562 jobs on average. Note that these three-year averages are the result of an array of decisions made by professionals, fluctuations in supply and demand, absenteeism and overtime of employees, and much more. These events can not be completely predicted nor replicated and thus we do not know how likely they were to happen, we only know that they did. Moreover, the performance of *Parts* at any moment in time is not independent of the performance of the moments before. As a result, we effectively only have one run to which we can compare our simulation results. We do not know how likely the occurrence of the run was, however since it is our only observation we should treat it like it is a typical run. Thus, the aim of the operational validation of our model is to find a model for which the KPIs of a typical run are equal to the three year averages from the historical data.

6.2.2 Model iterations

The simulation model is the result of an iterative model building process. In each iteration the conceptual and operational validity of the model were checked. Until the final iteration the operational validity turned out to be satisfactory. For each iteration of the model 25 runs with a duration of 30000 simulation time units (roughly 73000 jobs) were done. In each iterations assumptions had to be added, relaxed or combined. The first main iteration resulted in a base model of a job shop, in which jobs arrive at the shop, are released to the floor, are processed on one or more machines and depart. This is modeled similar to Algorithms 5.2, 5.3, 5.4, 5.6, and a basic version of Algorithm 5.8 from the final model.

However, when checking this iteration it became clear that a lot of hidden dynamics disturbed the regular flow of operations. The base model was not able to capture these dynamics and was therefore not suitable for further analysis. Therefore, disturbances were added to the model in the form of yield loss, machine breakdowns and external operations. These dynamics are mainly modeled in Algorithms 5.8, 5.9, 5.7, and 5.10 from the final model.

However, the addition of these disturbances resulted in a system in which some machines had an effective utilisation greater than one. After more analysis it became clear that overtime is one of the measures used to correct this. To model this, we implemented an overtime dynamic in third iteration. This overtime dynamic was based on the current queue length of a machine and an overtime rate based on the historic data and modelled by a basic version of Algorithm 5.5.

The model from the third iteration resulted in an average WIP closer to that observed at *Parts*. However, the WIP varied highly between the runs as well as during the runs. These fluctuations between the runs mean that it is hard to identify a typical run to compare to the empirical data. Also the high WIP fluctuations within the runs is something we do not observe in the historical data. We believe this arises from the fact that in reality an array of response time options is used by professionals to correct for high WIP. Trying to capture all of these interventions separately would not be mathematically tractable (de Kok et al., 2018). To solve this we changed the already implemented overtime mechanism such that it represents the aggregate effect of all separate interventions. This mechanism aims to captures all interventions done to increase productivity in case the available or planned work would exceed capacity by too much. This resulted in the final version of the model which differs from iteration three mostly in Algorithm 5.5.

Table 6.1 gives a brief overview of the four iterations, the main changes made in each iteration, and typical values for the KPIs of these iterations. This table illustrates that the KPI values changed between the different iterations and that the KPI values of the final iteration are close to the historical data. Additionally, the final model results in runs with less fluctuation in WIP similar to what we observed at *Parts*. Furthermore, the average value of the runs and the value of a typical runs with this model are very similar. Therefore we evaluate interventions tested with the model based on the average KPI values of multiple runs.

Table 6.1: Overview of model iterations

Iteration	Additions	Key Performance Indicators		
		Delivery Performance	Production Time	Work in Progress
1. Base		0.275	111.1	789
2. Disturbed	Yield loss and machine failures	0.115	115.5	2062
3. Basic overtime	Overtime based on queue length	0.126	114.1	1806
4. Advanced overtime	Overtime based on arriving work	0.145	114.4	1554
Actual situation		0.141	113.3	1562

Overview of additions made in key model iterations and the scores on key performance indicators in these iterations. Units of the key performance indicators are as follows: delivery performance is in percentage of jobs not tardy, production time is in simulation time units, and work in progress is in number of jobs.

6.3 Summary

This chapter discussed the validation and verification of the simulation model. The validation and verification consists of conceptual model validation, computerized model verification, and operational validation. The conceptual validation of the model was done by intensively discussing the conceptual model with domain experts. Next, the verification of the computerized model was done using traces, face validity, and comparison to other models. The results of these verification methods were positive and we were able to verify the computerized model. Finally, the operational validity of the model was checked by comparing important KPIs of the problem system with the results from the model. These comparisons were also positive, therefore it was concluded that the model is operationally valid.

Since we validated and verified the computerized model and have therefore presented a simulation model capable of representing the situation at *Parts*, we conclude that real-life high-mix low-volume job shops can be simulated by extending a basic job shop simulation model with disturbances and an aggregate mechanism modeling human interventions.

Part III
Solution design

In this Part, we present our analysis of the effectiveness of controlled order release and dispatching using the simulation model developed in Part II. This analysis includes the answers to SRQ2 and SRQ3. We introduce three main KPIs in the form of work in progress reduction, throughput increase, and overtime reduction and test the performance of different order release and machine dispatching rules using the simulation model.

Part III is organised as follows. In Chapter 7, we present the results of using different order release and dispatching methods starting by introducing the KPIs in Section 7.1. In Subsection 7.2.1 we present the results of machine dispatching rules while using the same release rule, IMM, as in the base model. Next, in Subsection 7.2.2, we again test these machine dispatching rules, now using MNJ as release rule. We continue by testing the PBB release rule in Subsection 7.2.3. Thereafter conclusions based on these results are presented in Chapter 8 which starts with a discussion of these results in Section 8.1. Afterwards we reflect on the research in this thesis in Section 8.2. Finally, recommendations for *Parts* based on the findings of this thesis are given in Section 8.3.

7 Results

In this chapter the effectiveness of shop floor control and order release measures are discussed. The simulation model as described in Chapter 5 is used to test different combinations of machine dispatching, pre-shop pool sequencing, and order release rules. This chapter starts with the introduction of the KPIs used in the evaluating of the measures in Section 7.1. Section 7.2 presents the results for these KPIs when using order release and shop floor control. This chapter ends with a summary in Section 7.3.

7.1 Key performance indicators

This chapter presents results to serve as evidence for the answers to the second and third sub research questions under investigation in this thesis and this section introduces the KPIs necessary to do this. Recall that these sub research questions are:

- **SRQ2**: How can dispatching improve control of a complex high-mix-low-volume job shop manufacturing environment by reducing work in progress inventory without reducing throughput?
- **SRQ3**: How can order release improve control of a complex high-mix-low-volume job shop manufacturing environment by reducing work in progress inventory without reducing throughput?

To successfully answer these SRQs it is necessary to know the resulting WIP and throughput for the different dispatching and order release rules. Therefore the first two KPI's reported in this section concern the average WIP on the shop floor and the average throughput. Since this throughput could be realized by varying amounts of overtime, we also report a third KPI concerning the time that was available to realize this throughput. Moreover, we are interested in how the different dispatching and release rules perform compared to the base model. For this reason, the KPIs are expressed relative to the base model. Each time these KPIs are tabulated, the absolute value of the underlying KPI is shown in round brackets for reference.

Note that a decrease in WIP and overtime are beneficial for performance while a decrease in throughput is disadvantageous for performance. For clarity reasons, we define our KPIs such that an increase in the KPI contributes to a positive effect on performance. Therefore we consider the following three KPIs in our study: WIP reduction compared to the base model in average number of jobs on the shop floor per time unit, throughput increase compared to the base model in average number of jobs per time unit, and available time reduction compared to the base model in time units. Where the base model uses immediate release (IMM) with first come first serve (FCFS) machine dispatching. All analyses are based on a time period corresponding to three years for which 25 replications have been done.

7.2 Order release and machine dispatching

This section presents the results of using order release and shop floor control using the simulation model. Subsection 7.2.1 presents the results of continuing with IMM in combination with different machine dispatching rules. The results of using maximum number of jobs and path based bottleneck as release rule are discussed in Subsections 7.2.2 and 7.2.3 respectively. In Subsection 7.2.4 we evaluate results under the assumption that a decrease in workload would increase efficiency.

7.2.1 Immediate release

In this section three sequence rules from the literature used in this thesis are formally introduced and their performance as machine dispatching rules are tested. Recall that all rules are compared to the base model, which uses IMM as release rule and FCF as machine dispatching rule. The first sequence rule EDD uses only information on the characteristics of the job. EDD sequences jobs based on their due date giving priority to the job with the earliest due date. Algorithm 7.1 gives a systematic representation.

Algorithm 7.1 Earliest due date

```
Require: $jobs$
1: $job_{current}$ is first job in $jobs$
2: $due_{current}$ is due date of $job_{current}$
3: for $job_{new}$ in $jobs$ do
4:     $due_{new}$ is due date of $job_{new}$
5:     if $due_{new} < job_{new}$ then
6:         $job_{current}$ is $job_{new}$
7:         $due_{current}$ is $due_{new}$
8:     end if
9: end for
10: return $job_{current}$
```

The second and third sequencing rules tested are LST and MCR, both rules consider job characteristics and the current simulation time. LST gives priority to the job with the least slack time, where slack time is the time until a job is due minus the processing time remaining for that job. MCR gives priority to the job with the minimum critical ratio, where the critical ratio is the time until a job is due divided by the processing time remaining for that job (see: Section 3.2). Equation 18 and Algorithm 7.2 and Equation 19 and Algorithm 7.3 show the calculations and systematic representations for LST and MCR respectively.

$$S_j = t_{now} - d_j - \sum_{m=1}^{n} P_{mj} \tag{18}$$

$$CR_j = \frac{t_{now} - d_j}{\sum_{m=1}^{n} P_{mj}} \tag{19}$$

where

S_j	= Slack of job j,
CR_j	= Critical ratio of job j,
t_{now}	= Current time,
d_j	= Due date of job j,
P_{mj}	= Processing time of job j at machine m,
n	= Number of machines in the shop

Algorithm 7.2 Least Slack Time

```
Require: $jobs$
1: $time_{now}$ is current simulation time
2: $job_{current}$ is first job in $jobs$
3: $due_{current}$ is due date of $job_{current}$
4: $process_{current}$ is remaining process time of $j_{current}$
5: $slack_{current} \leftarrow due_{current} - time_{now} - process_{current}$
6: for $job_{new}$ in $jobs$ do
7:     $job_{new}$ is first job in $jobs$
8:     $due_{new}$ is due date of $job_{new}$
9:     $process_{new}$ is remaining process time of $job_{new}$
10:    $slack_{new} \leftarrow due_{new} - time_{now} - process_{new}$
11:    if $slack_{new} < slack_{current}$ then
12:        $job_{current}$ is $job_{new}$
13:        $slack_{current}$ is $slack_{new}$
14:    end if
15: end for
16: return $job_{current}$
```

Algorithm 7.3 Critical Ratio

```
Require: $jobs$
 1: $time_{now}$ is current simulation time
 2: $job_{current}$ is first job in $jobs$
 3: $due_{current}$ is due date of $job_{current}$
 4: $process_{current}$ is remaining process time of $j_{current}$
 5: $cr_{current} \leftarrow (due_{current} - time_{now})/process_{current}$
 6: for $job_{new}$ in $jobs$ do
 7:     $job_{new}$ is first job in $jobs$
 8:     $due_{new}$ is due date of $job_{new}$
 9:     $process_{new}$ is remaining process time of $job_{new}$
10:     $cr_{new} \leftarrow (due_{new} - time_{now})/process_{new}$
11:     if $cr_{new} < cr_{current}$ then
12:         $job_{current}$ is $job_{new}$
13:         $cr_{current}$ is $cr_{new}$
14:     end if
15: end for
16: return $job_{current}$
```

The three introduced sequence rules, EDD, LST, and MCR, are tested as machine dispatching rules with IMM. Table 7.1 illustrates how the three dispatching rules perform compared to the base model. Note the big difference in WIP reduction between the three machine dispatching rules. This can be explained by the expected processing times of the jobs each rule prioritizes. As explained in Subsection 5.4.5, the time until a job is due upon arrival is positively correlated to the expected processing time of that job (i.e. on average jobs with shorter expected processing times have less time until they are due upon arrival than jobs with longer expected processing times). As a result EDD is a little biased towards jobs with shorter expected processing times, similar to the shortest processing time sequence rule (SPT). It is known that using SPT as machine dispatching rule minimizes the total WIP, and it is therefore unsurprising that in this case EDD as machine dispatching rule also leads to a large WIP reduction. LST prioritizes the jobs with the least slack and we know that the slack of jobs upon arrival is positively correlated to the expected processing time of that job (i.e. on average jobs with shorter expected processing times have less slack upon arrival than jobs with longer expected processing times). Therefore, LST is also biased towards shorter jobs and reduces WIP as well decreasing the expected WIP. The opposite is true for MCR, since the larger the processing time of a job, the lower the job's critical ratio (i.e. for two jobs with the same arrival time and due date MCR prioritizes the job with the largest expected processing time). As a result, MCR is biased towards jobs with high expected processing times increasing the expected WIP.

For all three machine dispatching rules an increase in throughput is found, however, the explanation for this increase in the case of EDD and LST is different from the explanation in case of MCR. As explained above EDD and LST are biased towards jobs with shorter processing times and hence resemble SPT. Therefore it is unsurprising that the expected throughput increases compared to the base situation in which FCF is used. The increase in throughput using MCR as machine dispatching rule has two different possible explanations, both a result of the increase in WIP when using MCR. Firstly, this increase in WIP can lead to higher machine utilisation and therefore an increased throughput and secondly, this increase in WIP can lead to a more overtime and therefore a higher throughput.

The final KPI under consideration is overtime, for which we see only small differences. EDD and LST result in a small reduction of overtime relative to the base model, while MCR results in the usage of a bit more overtime. This increase in overtime for MCR is very small, and therefore we conclude that the increase in throughput in case MCR is used it mostly due to the reduction of machine idleness.

After correcting for overtime the effective changes in throughput are 0.69%, 0.49%, and 1.10% for using EDD, LST and MCR as machine dispatching rule respectively. Thus using EDD and LST both reduce the WIP as well as increase the effective throughput. However, the constant prioritization of jobs with shorter processing times could be unsatisfactory if job profits are taken into account. Moreover, the WIP reductions from these dispatching rules are not high. Therefore, we conclude that although smart machine dispatching may improve the performance of *Parts*, currently these improvements are small and should not get priority.

Table 7.1: Immediate Release

	EDD		LST		MCR	
Work in progress reduction	7.40%	(1439.13)	1.22%	(1535.24)	-15.7%	(1797.48)
Throughput increase	0.69%	(7.30)	0.49%	(7.28)	1.12%	(7.33)
Overtime reduction	0.00%	(417.66)	0.01%	(417.65)	-0.01%	(417.73)

Percentage changes with respect to the base model when using immediate release for different machine dispatching rules with absolute values between round brackets for reference.

7.2.2 Maximum number of jobs

In this subsection we formally introduce the maximum number of jobs (MNJ) release rule and the performance of this rule is compared to the performance of the base rule, IMM. MNJ checks the number of jobs on the shop floor and releases a job in case this number is below a threshold. In the model this is implemented by checking for every job in the system whether it is currently queuing (its status is que) or being processed (its status is pro). If this is the case, this job is part of the current work in progress on the shop floor. In case this work in progress on the shop floor does not exceed a threshold a new job can be released. A systematic representation of this rule is given in Algorithm 7.4.

Algorithm 7.4 Maximum number of jobs

```
Require: jobs
1: $wip \leftarrow 0$
2: for job in jobs do
3:     if state of job is que or pro then
4:         $wip = wip + 1$
5:     end if
6: end for
7: if $wip < threshold$ then
8:     return True
9: else
10:    return False
11: end if
```

MNJ can be used with a wide variety of pre-shop pool sequencing rules. Here we test the performance of MNJ using the three sequence rules introduced in Subsection 7.2.1 and the basic sequence rule FCF as pre-shop pool sequencing rules. Also we test every pre-shop pool sequence rule in combination with the same four sequence rules as machine dispatching rules. The maximum number of jobs threshold is set to 1250, which is 20% lower than the current average. Tables 7.2, 7.3, and 7.4 show the results of each combination of sequence and dispatch decision for work in progress reduction, throughput increase, and overtime reduction respectively.

Consider the decrease in WIP for different combinations of pre-shop pool sequence and machine dispatching rules presented in Table 7.2. For each combination of pre-shop pool sequence and machine dispatching rules the work in progress is reduced significantly, with a minimum reduction (25.15%) realised in WIP for the combination of EDD as pre-shop pool sequence and MCR as machine dispatching rules respectively. In this situation, two subsequent jobs are expected to have similar due dates as a result of the EDD pre-shop pool sequencing rule. When these two jobs are compared on the shop floor priority will be given based on the MCR machine dispatching rule. The critical ratio of two jobs with the same due date is lowest for the job with the highest processing time and as a result the decrease in WIP as a result of MNJ release rules is smaller.

The largest decrease in WIP (30.74%) is realized in the situation where the pre-shop pool is sequenced according to LST and machine dispatching is done by EDD. Here two subsequent released jobs are expected to have similar slack as a result of the LST pre-shop pool sequence rule. In case these two jobs are compared on the shop floor according to the EDD machine dispatching rule, priority thus will be given to the job with the lowest expected processing time. As a result the WIP is further reduced utilizing this combination of pre-shop pool sequence and machine dispatch rules.

The average WIP decrease for each distinct pre-shop pool sequencing method is maximum for LST and minimum for MCR. However, the difference between these two averages is only 1.8%. Moreover, the average WIP decrease for each distinct machine dispatching method is maximum for EDD and minimum for FCF with a difference between these two averages of 0.8%. Meaning that the combination of the sequence rules that perform best on average as pre-shop pool sequencing and the machine dispatching rule, also yields the highest overall WIP reduction.

It is not unexpected that machine dispatching following EDD results in the largest reduction of WIP due to the relationship between jobs due dates and expected processing times explained in Subsection 7.2.1. The time until a job is due upon arrival is positively correlated to the expected processing time of that job (i.e. on average jobs with shorter expected processing times have less time until they are due upon arrival than jobs with longer expected processing times). As a result EDD is a little biased towards jobs with shorter expected processing times, similar to SPT. It is know that SPT minimizes the total WIP, and it is therefore unsurprising that in this case EDD as machine dispatching again results in the largest WIP reduction.

Table 7.2: Maximum Number of Jobs - Work in progress reduction

		Machine dispatching							
		FCF		EDD		LST		MCR	
Pre-shop pool sequencing	FCF	26.16%	(1147)	28.18%	(1116)	28.72%	(1107)	29.81%	(1091)
	EDD	28.80%	(1106)	27.64%	(1124)	27.81%	(1121)	25.15%	(1163)
	LST	28.05%	(1118)	30.74%	(1076)	28.60%	(1109)	29.02%	(1103)
	MCR	26.83%	(1137)	26.64%	(1140)	27.68%	(1124)	27.94%	(1119)

Percentage work in progress reduction with respect to the base model when using maximum number of jobs for different machine dispatching rules and pre-shop pool sequencing rules with absolute values between round brackets for reference.

The second KPI under consideration is the throughput increase. Note that each combination of pre-shop pool sequence and machine dispatching rule with MNJ order release results in a negative increase and thus a decrease of the throughput. This is expected, since using MNJ the maximum work in progress is limited increasing the probability a machine becomes idle. Hence, the utilisation of the machines could go down leading to a decrease in throughput. The pre-shop pool sequencing rule and machine dispatching rules which on average lead to the smallest throughput reduction are FCF and MCR respectively. Also, the combination of these two rules leads to the smallest overall reduction in throughput. The fact that MCR as machine dispatching rule performs this well can be linked to the small reduction in WIP this rule on average yields as explained above. However, as pre-shop pool sequence rule MCR performs the worst of all sequence rules tested.

Table 7.3: Maximum Number of Jobs - Throughput increase

		Machine dispatching							
		FCF		EDD		LST		MCR	
Pre-shop pool sequencing	FCF	-4.01%	(6.96)	-2.35%	(7.08)	-2.38%	(7.08)	-1.08%	(7.17)
	EDD	-2.46%	(7.07)	-2.55%	(7.06)	-2.47%	(7.07)	-4.42%	(6.93)
	LST	-2.27%	(7.08)	-3.00%	(7.03)	-3.44%	(7.00)	-2.92%	(7.04)
	MCR	-4.97%	(6.89)	-5.19%	(6.87)	-5.47%	(6.85)	-4.10%	(6.95)

Percentage throughput increase with respect to the base model when using maximum number of jobs for different machine dispatching rules and pre-shop pool sequencing rules with absolute values between round brackets for reference.

The final KPI under consideration is the reduction in overtime used. Table 7.4 shows a decrease in overtime used for all combinations of pre-shop pool sequence and machine dispatching rules. On average the best performing pre-shop pool sequencing rule is LST and the best performing machine dispatching rule EDD. Recall that this is also the case for WIP reduction. For this KPI however the combination of the on average best performing pre-shop pool sequence and machine dispatching rules does not yield the best overall result. In this case the best overall result is achieved by combining MCR and EDD as pre-shop pool sequence and machine dispatching rule respectively.

The changes in throughput and overtime should not be seen as independent since the increase (decrease) of overtime for the same throughput corresponds to a decrease (increase) in effective throughput. Where effective throughput is the throughput corrected for the change in available time as a result of overtime. The effective throughput increased for all combinations of pre-shop pool sequencing and machine dispatching are tabulated in Table 7.5.

Note the effective throughput decreases for all combinations of pre-shop pool sequencing and machine dispatching when using MNJ. This most likely is due to the fact that the MNJ stops releasing all jobs in case the WIP exceeds the threshold which could lead to high WIP at some machines and idleness of others. At *Parts* this effect is magnified due to the fact that jobs could be separated based on the type of starting material and the routings of these two types of jobs are unlikely to overlap significantly. Also, since MNJ does not take detailed information on the shop into account, the observed decrease in throughput is in line with the conclusion of Betrand and Van Ooijen (2008). They noted that when there are no workload dependent server rates TTT can only reduce if a release rule takes detailed information about the shop status into account.

Table 7.4: Maximum Number of Jobs - Overtime decrease

		Machine dispatching							
		FCF		EDD		LST		MCR	
Pre-shop pool sequencing	FCF	0.27%	(416.56)	0.11%	(417.21)	0.07%	(417.36)	0.08%	(417.32)
	EDD	0.14%	(417.10)	0.06%	(417.40)	0.01%	(417.63)	0.14%	(417.10)
	EDD	0.11%	(417.22)	0.17%	(416.97)	0.16%	(417.00)	0.16%	(417.00)
	MCR	0.11%	(417.19)	0.30%	(416.40)	0.11%	(417.19)	0.01%	(417.63)

Percentage overtime reduction with respect to the base model when using maximum number of jobs for different machine dispatching rules and pre-shop pool sequencing rules with absolute values between round brackets for reference.

Table 7.5: Maximum Number of Jobs - Effective throughput increase

		Machine dispatching			
		FCF	EDD	LST	MCR
Pre-shop pool sequencing	FCF	-3.76%	-2.24%	-2.30%	-1.00%
	EDD	-2.33%	-2.49%	-4.46%	-4.29%
	EDD	-2.17%	-2.84%	-3.29%	-2.76%
	MCR	-4.87%	-4.91%	-5.36%	-4.09%

Percentage effective throughput increase with respect to the base model when using maximum number of jobs for different machine dispatching rules and pre-shop pool sequencing rules.

7.2.3 Path based bottleneck

The second release rule compared to IMM is the PBB (*Path based bottleneck* see: Section 3.1). PBB only releases those jobs for which releasing them will not exceed a load threshold for any machine in their routing. Where the load of a machine is the sum of expected processing times on a machine for all jobs released. Thus for each machine in the routing of a job it is checked whether the current load plus the load the job would add is below the load threshold for that machine. If so, the job is eligible for release. PBB checks whether the first job could be released without exceeding the load threshold of of all machines on that job's routing. If this is possible the job is released and otherwise the next job is checked. A systematic representation is given in Algorithm 7.5. Since this release rule checks the load of all machines separately opposed to MNJ which considers the shop as a whole, this rule should result in less utilisation loss and thus less throughput loss.

Algorithm 7.5 Path based bottleneck

```
Require: $jobs, threshold$
1: for $job$ in $jobs$ do
2:     $routing$ is routing of $job$
3:     for $operation$ in $routing$ do
4:         $process$ is process time of $operation$
5:         $machine$ is machine of $operation$
6:         $load$ is current load of $machine$
7:         if $load - process \leq threshold$ then
8:             return $job$
9:         end if
10:    end for
11: end for
12: return False
```

PBB takes a list of jobs sorted by the minimum slack ratio (MSR) sequence rule as input. MSR prioritizes the job with the minimum slack ratio. The slack ratio aims to find the average proportion of slack of all the machines visited by a job which the job would consume upon release (Philipoom et al., 1993). The slack ratio for a job can be calculated as shown in Equation 20. This is implemented in the simulation model as illustrated systematically in Algorithm 7.6.

$$SR_j = \frac{\sum_{i=1}^{n} \frac{P_{mj}}{T-L_m}}{N_j} \tag{20}$$

where

SR_j	= Slack ratio of job j,
P_{mj}	= Processing time of job j at machine m,
T	= Capacity threshold,
L_m	= Current total load at machine m,
N_j	= Number of operations required by job j, and
n	= Number of machines in the shop

Algorithm 7.6 Slack Ratio

```
Require: $jobs, capacity$
 1: $job_{current}$ is first job in $jobs$
 2: $routing_{current}$ is routing of $job_{current}$
 3: $number_{current}$ is length of $routing_{current}$
 4: for $operation$ in $routing_{current}$ do
 5:     $process$ is process time of $operation$
 6:     $machine$ is machine of $operation$
 7:     $load$ is current load of $machine$
 8:     $ratio_{current} \leftarrow ratio_{current} + process/(capacity - load)$
 9: end for
10: $ratio_{current} \leftarrow ratio_{current}/number_{current}$
11: for $job_{new}$ in $jobs$ do
12:     $job_{new}$ is first job in $jobs$
13:     $routing_{new}$ is routing of $job_{new}$
14:     $number_{new}$ is length of $routing_{new}$
15:     for $operation$ in $routing_{new}$ do
16:         $process$ is process time of $operation$
17:         $machine$ is machine of $operation$
18:         $load$ is current load of $machine$
19:         $ratio_{new} \leftarrow ratio_{new} + process/(capacity - load)$
20:     end for
21:     $ratio_{new} \leftarrow ratio_{new}/number_{new}$
22:     if $ratio_{new} < ratio_{current}$ then
23:         $job_{current}$ is $job_{new}$
24:         $ratio_{current}$ is $ratio_{new}$
25:     end if
26: end for
27: return $job_{current}$
```

PBB can be combined with with a wide variety of pre-shop pool sequencing rules. Similar to the analysis for MNJ, we test the performance of PBB using the three sequence rules introduced in Subsection 7.2.1 and the basic sequence rule FCF as machine dispatching rules. To compare the performance of PBB in combination with different machine dispatching rules we test each combination when the PBB threshold is set to 350. This threshold is set after several short exploratory simulation runs. Table 7.6 reports how the four dispatching rules perform compared to the base model.

When using PBB a reduction in WIP, throughput, and overtime is achieved regardless of the machine dispatching rule used. The largest WIP and smallest throughput reductions are achieved when using LST as machine dispatching rule and the largest overtime reduction when MCR is used.

Table 7.6: Path Based Bottleneck

	FCF		EDD		LST		MCR	
Work in progress reduction	34.48%	(1021)	35.54%	(1001)	38.63%	(953)	25.06%	(1164)
Throughput increase	-1.86%	(7.11)	-0.94%	(7.18)	-0.69%	(7.20)	-1.18%	(7.16)
Overtime reduction	0.06%	(417.4)	-0.06%	(417.4)	0.09%	(417.3)	0.11%	(417.2)

Percentage changes with respect to the base model when using path based bottleneck for different machine dispatching rules with absolute values between round brackets for reference.

To further investigate the effectiveness of PBB on the KPIs we test PBB for different threshold values. Since LST and MCR were the best performing machine dispatching rules for a threshold of 350, we use these rules also for this test. Figures 7.1 and 7.2 show how PBB performs for different threshold values with MCR and LST as machine dispatching rules respectively. These figures illustrate that a trade-off between maximizing the decrease in WIP and overtime and minimizing the decrease in throughput. We do not model the increase in effectiveness resulting form decreasing WIP (workload dependent service rates). Consequently, it is logical the results show this trade-off. By means of comparing the results (illustrated in the figures below) we noted that combining PBB with LST machine dispatching rule results in a greater decrease in WIP for a smaller decrease in throughput than combining PBB with MCR as machine dispatching rule.

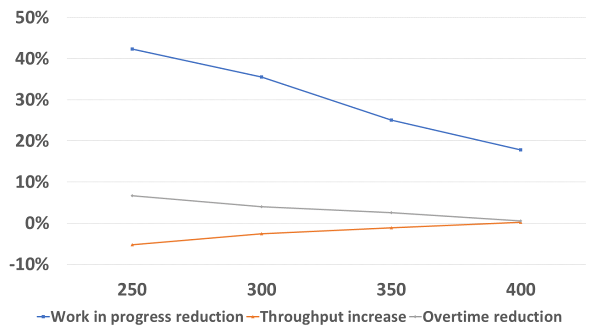

Figure 7.1: Path based bottleneck with critical ratio dispatching for different threshold values

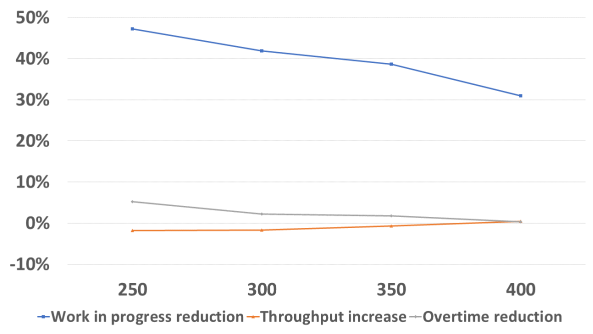

Figure 7.2: Path based bottleneck with least slack dispatching for different threshold values

7.2.4 Increased effectiveness

In our model we did not include the effects of workload dependent service rates. However, in Subsection 2.1.2 we provided evidence from literature for the existence of such an effect. In the situation where PBB with a machine load threshold of 350 is combined with LST machine dispatching we observed a decrease in WIP of 37%. This type of WIP reduction should lead to an increase in efficiency following the work efficiency smiley introduced in Section 2.1, therefore we test how this combination of order release and machine dispatching rules would perform under the assumption that the efficiency would increase as a result of the WIP reduction. Empirical results showed a 10% decrease in WIP resulted in a 1.0% efficiency increase after a year (Bertrand and Van Ooijen, 2002). We therefore assume that the 38% reduction of WIP would result in a efficiency increase between 1.0% and 3.7%. The WIP and effective throughput values when using PBB with a load threshold of 350 and LST machine dispatching for overall efficiency increases of 1.0% and 3.7% are tabulated in Table 7.7. These results show that using PBB order release in combination with LST machine dispatching can lead to a WIP decrease between 40.65% and 45.16% and a throughput increase between 0.19% and 0.93%. Therefore, we conclude that order release can improve shop performance by reducing WIP levels at machines while preventing large increased in idleness if there is an efficiency gain from the reduced work in progress.

Table 7.7: Path Based Bottleneck with increased efficiency

	1.0% efficiency increase	3.7% efficiency increase
Work in progress reduction	40.65%	45.16%
Effective throughput increase	0.19%	0.93%

Percentage changes with respect to the base model when using PBB with a load threshold of 350 and LST machine dispatching for overall efficiency increases of 1.0% and 3.7%.

7.3 Summary

This chapter presented the results of using workload control through controlled order release and dispatching. We introduced three main KPIs in the form of work in progress reduction, throughput increase, and overtime reduction and tested the performance of different order release and machine dispatching rules using the simulation model. We concluded that although smart machine dispatching can improve the performance of *Parts*, currently these improvements are small and should not get priority. However, with workload control through controlled order release it is possible to strongly decrease WIP while only slightly decreasing throughput even when the efficiency gains of this decreased WIP are not considered. Finally, we included the expected efficiency gain resulting for reducing the work in progress into our tests. The performed tests showed that a high reduction of work in progress can be realised while increasing throughput by using controlled order release in the form of path based bottleneck.

8 Conclusion

The objective of this thesis was to investigate the possible improvement of production performance of *Parts* through implementing controlled order release and machine dispatching. For this purpose, current disturbing factors at *Parts* were identified, and thereafter possible machine dispatching and order release rules tested using an empirically valid simulation model. This final chapter draws conclusions on the insights obtained in this thesis and reflects on the research conducted. Section 8.1 summarizes the conclusions of this thesis. After which, we formulate directions for further research based on insights obtained from this thesis and possible improvements to the research in Section 8.2.2. This thesis ends with an overview of recommendations for *Parts* in Section 8.3.

8.1 Conclusions

Prior to the start of this research the dynamics of daily planning within the job shop of *Parts* were not fully understood which led to little control on production. In an attempt to improve control, *Parts* increased safety times and started expediting delayed jobs by giving them priority over other jobs. The implementation of safety times increases work in progress inventory and the expediting of jobs creates additional nervousness. As a result, *Parts* finds itself in a downwards spiral with high work in progress inventory and low control on production.

We identified reducing the work in progress as a possible method to break this spiral and hopefully improve the performance of *Parts*. We provided evidence from literature suggesting a negative relation between high work in progress and performance, further strengthening the proposition that reducing work in progress is a suitable method to improve the performance of *Parts*. However, reducing work in progress inventory can lead to machine idleness resulting in a reduction in throughput. Thus we formulated the following research question aiming to improve the performance of *Parts* by reducing the work in progress without reducing the throughput:

How to control order release and dispatching in a complex high-mix-low-volume job shop manufacturing environment to reduce work in progress inventory without reducing throughput?

To answer this (main) research question a simulation study was set-up and three sub research questions have been formulated. Combining the answers to these sub research questions provides the necessary insights for answering the main research question. Recall the first sub research question:

- **SRQ1**: How can a complex high-mix-low-volume job shop manufacturing environment be simulated?

In this thesis we presented a model which is the result of iterative model building process. This process consisted of redefining, testing, and validating both conceptual and computerized models on the basis of weekly interviews and empirical data. The model extends the basic structure of a job shop simulation model with three disturbances (yield loss, machine breakdowns, and external operations). In addition to this, we included an overtime mechanism to model the aggregate effects of human interventions that are typically not considered in literature. We demonstrated that inclusion of the disturbances and the overtime mechanism resulted in an empirically valid model, in the sense that typical simulations of the resulting model are in line with historical data from *Parts*. We therefore concluded that real-life high-mix low-volume job shops can be simulated by extending a basic job shop simulation model with disturbances and an aggregate mechanism modeling human interventions. The second sub research question concerns the effectiveness of machine dispatching on increasing shop performance.

- **SRQ2**: How can dispatching improve control of a complex high-mix-low-volume job shop manufacturing environment by reducing work in progress inventory without reducing throughput?

In this thesis we compared the performance of three machine dispatching methods to that of first come first serve used in the base model. The three machine dispatching methods are: earliest due date, least slack time, and critical ratio. For earliest due date and least slack time we found an increase in throughput and a decrease in work in progress. This can be attributed to the fact that using these rules for machine dispatching will prioritize jobs with shorter processing times. However, continuously prioritizing these jobs might not be a suitable strategy in case the profits of these jobs are included as well. For critical ratio we also found an increase in throughput, however this time paired with a increase in work in progress. Critical ratio increased work in progress by prioritizing working on the jobs with longer processing times. This increase in work in progress reduces the expected idleness making up for the throughput lost by prioritizing these jobs. We concluded that although smart machine dispatching can improve the performance of *Parts*, currently these improvements are small and should not get priority. The third, and final, sub research question concerns the effectiveness of controlled order release on shop performance.

- **SRQ3**: How can order release improve control of a complex high-mix-low-volume job shop manufacturing environment by reducing work in progress inventory without reducing throughput?

We tested the effectiveness of two order release rules discussed in the literature using our simulation model. The first order release rule tested is maximum number of jobs which limits the work in progress on the shop floor to a maximum. We found that this rule results in a decrease in work in progress. However this decrease in work in progress leads to increased idleness of machines resulting in a significant decrease in throughput.

Aiming to reduce the increase in idleness as a consequence of limiting the work in progress, we tested the effectiveness of the path based bottleneck release rule. The path based bottleneck release rule only releases those jobs for which releasing them will not exceed a load threshold for any machine in their routing. The results of these tests showed that large work in progress reductions with only small reductions in throughput can be achieved by path based bottleneck controlled order release. These results are achieved without considering the increased effectiveness as a result of decreasing work in progress. As illustration of the additional potential benefits when considering the expected efficiency gain resulting for reducing the work in progress, we include this effect into our final tests. These tests showed that a high reduction of work in progress can be realised while simultaneously increasing throughput by using controlled order release in the form of path based bottleneck.

In short, we showed that real-life high-mix low-volume job shops can be simulated by extending a basic job shop simulation model with disturbances and an aggregate mechanism modeling human interventions. And showed that implementing a pre-shop pool will lead to a very substantial reduction in work in progress inventory on the shop floor, and thus to a reduction in working capital. Moreover, by combining our insights with literature we may expect that a pre-shop pool will increase the throughput of the shop, reduce the number of manual interventions once parts are released onto the shop floor, and improve worker satisfaction and retention.

8.2 Reflection

In this section we reflect on the research in this thesis. First, we describe several contributions of this thesis to the literature. Thereafter, several limitations of this study are discussed. Finally, we propose future research directions to investigate the effects of these limitations and to build further upon the findings of this thesis.

8.2.1 Contribution to literature

This thesis contributes to the literature in two ways. The developed job shop simulation model extends the basic simulation model in several aspects. The model includes the disturbing effects of yield loss, machine breakdowns, and external operations. Also, it includes a mechanism to model the aggregate effects of human intervention that are typically not considered in literature. We showed the model is empirically valid, in the sense that typical simulations of the resulting model are in line with observations for a real-life job shop.

Furthermore, we tested the path based bottleneck release rule in a simulation model of a real-life job shop. We provide evidence that this rule can lead to large work in progress reductions with nearly no throughput loss. Earlier research only provided such evidence using simulation models of fictional job shops.

8.2.2 Limitations

The research in this thesis is subject to several limitations. Firstly, we conducted a literature study with the objective to identify machine dispatching and order release rules (control mechanisms) discussed in the literature. It turned out that a wide variety of control mechanisms have been introduced and both empirically tested and investigated through simulation. Given the large amount of methods available we cannot be certain that all have been included.

Furthermore, the results obtained with respect to machine dispatching rules are not easy to interpret in the context of this study since key job characteristics are out of scope. The profit of a job is such a characteristic, since this can strongly influence whether it is profitable to delay one job in favour of others or not. Since this is left out of scope, interpreting the effects of machine dispatching rules on business value is difficult.

Finally, our simulation model assumes a continuously working job shop (i.e. one that does not distinguish between working and closing hours or workdays and holidays). In order to still model the difference between the working hours available to the different machines, machine service rates have been adjusted in line with earlier studies. In the final iterations of the model it became clear that for a precise modeling of *Parts* it was also needed to incorporate an overtime mechanism. Due the the design choice of modeling differences in available working hours by adjusting the service rates on an individual machine bases, this also had to be modeled by further adjusting the service rates. Due to the inclusion of this types of overtime adjusted service rates, we were not able to include workload dependent service rates. As a result, the precise internal dynamics between machine specific workload and throughput could not be determined and we had to make job shop wide conclusions.

8.2.3 Future research

Several directions for future research follow from this study either by improving on the current limitations or by building upon the findings of this thesis. As mentioned more release rules have been introduced in literature than tested in this thesis. It would be interesting to see how other rules perform compared to the base model and the extensions we tested. For release rules specifically, it would be interesting to see how rules with probabilistic or time bucketing workload accounting would perform as they might reduce the probability of idleness even further.

Furthermore, it would be interesting to model the difference in available hours to machines and the difference between working and closing hours more accurately to analyse the internal dynamics between local workload decreases and throughput more precisely.

Moreover, sequence dependent setup times could be included into the model. Within *Parts* there are machines for which the setup times are strongly dependent on the sequence in which jobs are processed and investigating how this could be included in the order release rule would be very interesting. Earlier research has been conducted on this relation, however these studies did not include the disturbances this thesis did.

The expected effects of the work efficiency smiley in this thesis are based on empirical evidence from the literature. However, as seen from the JD-R model the effects of job demands on performance is buffered by job resources. Hence the precise effects of work load on efficiency can vary greatly across situations. Therefore, it would be interesting to further investigate this relation in different situations. The goal of these studies would be the development of a robust model linking the effects of work in progress to efficiency possibly including the effects of buffers.

Finally, the model in this thesis is developed to represent the situation at *Parts*. However, it would be interesting to see of the model parameters could be tuned in such a way to accurately represent other job shops as well. If this is the case, standard extensions (disturbances and aggregate human intervention mechanisms) on basic job shop simulation models to be able to quickly create a model for each new job shop.

8.3 Recommendations

We end this thesis with a number of recommendations to *VDL ETG* to improve performance and create business value. These recommendations are based on the results of the research in this thesis and insights obtained during the project.

- Implement a pre-shop pool to regulate the flow of incoming jobs to the shop floor. This study showed that large work in progress reductions are possible with little decrease in throughput by using path based bottleneck workload control when not considering the effectiveness gains of the workload reduction and even increase in throughput when these effectiveness gains are considered. Therefore our first recommendation is to start using workload control to reduce the workload within and thus improve the performance of *Parts*. However, we saw that using maximum number of jobs as order release rule increased machine idleness due to the distinct types of routings for jobs of different starting materials. Therefore, it is important to implement an order release rule which considers the capacity of machines separately instead of the capacity of the shop floor as a whole, path based bottleneck is such a rule.

- Use the the simulation model build in this thesis as basis for future research. The simulation model build in this study is a valid and verified model of *Parts*. Therefore it can be in the investigation into the effectiveness of interventions other than workload control as well. However, it is important to note that the model was build with the aim to test workload control and could be unsuitable to test very different interventions. We trust on the critical mind set of future researches to make well-advised decisions with regard to this. To assist future research the simulation model will be made available for *VDL ETG* including a manual and overview of assumptions and modeling decisions.

- Capture and collect more reliable data on all business processes. The more accurate and reliable data is available, the more accurate and precise the predictions based on this data are. Unfortunately reliable data was not always available and estimations had to be made based on interviews. More and preciser collection and storage of data can make more precise predictions possible.

- Investigate the effects of workload on efficiency for different machines. Based on the literature we established that a relation between workload and efficiency exists in the form of an inverted u-shape. However the workload corresponding to optimal efficiency (optimal workload) and the precise manner in which efficiency decreases when deviating from this optimum can vary greatly across situations. To improve the effectiveness of workload control in the form of order release it is necessary to know more on the optimal workload across machines. Therefore a research should be set up to further investigate this.

- Investigate the maximum fraction of jobs to expedite. To improve delivery performance *Parts* assigns priorities to jobs, giving, on average, precedence to 32% jobs at any moment in time. By trying to expedite this many jobs, other jobs are delayed. In the current situation these delayed are often expedited later on. This results in a lot of nervousness which is detrimental to the performance. *VDL ETG* should investigate this process of expediting jobs and create a formal data driven policy for prioritizing.

To conclude, this thesis showed that although a job shop manufacturing environment can be complex, it is possible to make data driven models to identify improvement possibilities. Specifically, we identified that implementation of controlled order release is likely to improve performance by significantly reducing the work in progress. We are glad that VDT ETG is acting on this insight and has created a project team to investigate the implementation of such a mechanism. We believe that by continuing to use a data driven approach with respect to this implementation, the implementation will be successfully and result in a substantial increase in business value.

References

A. Abu-Suleiman, D. B. Pratt, and B. Boardman. The modified critical ratio: towards sequencing with a continuous decision domain. *International Journal of Production Research*, 43(15):3287–3296, 2005.

S. C. Aggarwal, F. Paul Wyman, and B. A. McCarl. An investigation of a cost-based rule for job-shop scheduling. *International Journal of Production Research*, 11(3):247–261, 1973.

I. Ahmed and W. W. Fisher. Due date assignment, job order release, and sequencing interaction in job shop scheduling. *Decision sciences*, 23(3):633–647, 1992.

C. Baker and B. P. Dzielinski. Simulation of a simplified job shop. *Management Science*, 6(3):311–323, 1960.

K. R. Baker. The effects of input control in a simple scheduling model. *Journal of Operations Management*, 4(2):99–112, 1984.

K. R. Baker and J. J. Kanet. Job shop scheduling with modified due dates. *Journal of Operations Management*, 4(1):11–22, 1983.

K. R. Baker and D. Trietsch. *Principles of sequencing and scheduling*. John Wiley & Sons, 2013.

W. Bechte. Theory and practice of load-oriented manufacturing control. *The International Journal of Production Research*, 26(3):375–395, 1988.

D. Bergamaschi, R. Cigolini, M. Perona, and A. Portioli. Order review and release strategies ina job shop environment: A review and a classification. *International Journal of Production Research*, 35(2):399–420, 1997.

W. L. Berry, R. J. Penlesky, and T. E. Vollmann. Critical ratio scheduling: dynamic due-date procedures under demand uncertainty. *IIE transactions*, 16(1):81–89, 1984.

J. M. Bertrand and H. Van Ooijen. Workload based order release and productivity: a missing link. *Production Planning & Control*, 13(7):665–678, 2002.

J. W. M. Bertrand and J. C. Wortmann. *Production control and information systems for component-manufacturing shops*. Citeseer, 1981.

J. Betrand and H. Van Ooijen. Optimal work order release for make-to-order job shops with customer order lead-time costs, tardiness costs and work-in-process costs. *International Journal of Production Economics*, 116(2):233–241, 2008.

P. M. Bobrowski. Implementing a loading heuristic in a discrete release job shop. *The International Journal of Production Research*, 27(11):1935–1948, 1989.

R. J. Boucherie and N. M. Van Dijk. *Queueing networks: a fundamental approach*, volume 154. Springer Science & Business Media, 2010.

A. Brown, S. Dimitrov, and A. Y. Barlatt. Routing distributions and their impact on dispatch rules. *Computers & Industrial Engineering*, 88:293–306, 2015.

R. G. Brown. Simulations to explore alternative sequencing rules. *Naval Research Logistics Quarterly*, 15 (2):281–286, 1968.

M. H. Bulkin, J. L. Colley, and H. W. Steinhoff Jr. Load forecasting, priority sequencing, and simulation in a job shop control system. *Management Science*, 13(2):B–29, 1966.

R. Cigolini, M. Perona, and A. Portioli. Comparison of order review and release techniques in a dynamic and uncertain job shop environment. *International Journal of Production Research*, 36(11):2931–2951, 1998.

R. W. Conway. An experimental investigation of priority assignment in a job shop. Technical report, RAND CORP SANTA MONICA CALIF, 1964.

R. W. Conway, W. L. Maxwell, and L. W. Miller. *Theory of scheduling.* Courier Corporation, 2003.

T. de Kok et al. Inventory management: Modeling real-life supply chains and empirical validity. *Foundations and Trends® in Technology, Information and Operations Management*, 11(4):343–437, 2018.

E. Demerouti and A. B. Bakker. The job demands-resources model: Challenges for future research. *SA Journal of Industrial Psychology*, 37(2):01–09, 2011.

D. M. Diamond, A. M. Campbell, C. R. Park, J. Halonen, and P. R. Zoladz. The temporal dynamics model of emotional memory processing: a synthesis on the neurobiological basis of stress-induced amnesia, flashbulb and traumatic memories, and the yerkes-dodson law. *Neural plasticity*, 2007, 2007.

S. Eilon and D. Cotterjll. A modified si rule in job shop scheduling. *The International Journal of Production Research*, 7(2):135–145, 1968.

A. Fadil, F. Chetouane, and Z. Binder. Control of flexible job-shop: Disturbances and robustness. *IFAC Proceedings Volumes*, 30(19):13–18, 1997.

L. D. Fredendall, D. Ojha, and J. W. Patterson. Concerning the theory of workload control. *European Journal of Operational Research*, 201(1):99–111, 2010.

T. D. Fry and A. E. Smith. A procedure for implementing input output control a case st. *Production and Inventory Management Journal*, 28(4):50, 1987.

J. S. Fryer. Operating policies in multiechelon dual-constraint job shops. *Management Science*, 19(9): 1001–1012, 1973.

R. Germs. *Controlling the order pool in make-to-order production systems.* University of Groningen, 2012.

M. Hamidi. Two new sequencing rules for the non-preemptive single machine scheduling problem. *The Journal of Business Inquiry*, 15(2):116–127, 2016.

R. Hollier. A simulation study of sequencing in batch production. *Journal of the Operational Research Society*, 19(4):389–407, 1968.

O. Holthaus. Scheduling in job shops with machine breakdowns: an experimental study. *Computers & industrial engineering*, 36(1):137–162, 1999.

O. Holthaus and H. Ziegler. Look ahead job demanding for improving job shop performance. *Operations-Research-Spektrum*, 19(1):23–29, 1997.

C. H. Jones. An economic evaluation of job shop dispatching rules. *Management Science*, 20(3):293–307, 1973.

K. K. Kumar, D. Nagaraju, S. Gayathri, and S. Narayanan. Evaluation and selection of best priority sequencing rule in job shop scheduling using hybrid mcdm technique. In *IOP Conference Series: Materials Science and Engineering*, volume 197, page 012059. IOP Publishing, 2017.

M. J. Land and G. J. Gaalman. The performance of workload control concepts in job shops: Improving the release method. *International Journal of Production Economics*, 56:347–364, 1998.

A. M. Law, W. D. Kelton, and W. D. Kelton. *Simulation modeling and analysis*, volume 3. McGraw-Hill New York, 2000.

O. Lünsdorf and S. Scherfke. Simpy, discrete event simulation for python [software]. `https://simpy.readthedocs.io/en/latest/about/license.html`, 2013.

M. K. Malhotra, J. B. Jensen, and P. R. Philipoom. Management of vital customer priorities in job shop manufacturing environments. *Decision Sciences*, 25(5-6):711–736, 1994.

S. A. Melnyk and G. L. Ragatz. Order review/release: research issues and perspectives. *The International Journal Of Production Research*, 27(7):1081–1096, 1989.

S. A. Melnyk, D. R. Denzler, and L. Fredendall. Variance control vs. dispatching efficiency. *Production and Inventory Management Journal*, 33(3):6, 1992.

M. R. Moreira, R. Alves, et al. A new input-output control order release mechanism: how workload control improves manufacturing operations in a job shop. *Faculdade De Economia Working Papers, Research Paper Series-no*, 107:1–19, 2006.

M. R. A. Moreira and R. A. F. Alves. A methodology for planning and controlling workload in a job-shop: a four-way decision-making problem. *International Journal of Production Research*, 47(10):2805–2821, 2009.

M. Oral and J.-L. Malouin. Evaluation of the shortest processing time scheduling rule with truncation process. *AIIE Transactions*, 5(4):357–365, 1973.

S. S. Panwalkar and W. Iskander. A survey of scheduling rules. *Operations research*, 25(1):45–61, 1977.

M. Perona and A. Portioli. An enhanced loading model for the probabilistic workload control under workload imbalance. *Production planning & control*, 7(1):68–78, 1996.

P. Philipoom and D. Steele. Shop floor control when tacit worker knowledge is important. *Decision Sciences*, 42(3):655–688, 2011.

P. R. Philipoom and T. D. Fry. Order review/release in the absence of adherence to formal scheduling policies. *Journal of Operations Management*, 17(3):327–342, 1999.

P. R. Philipoom, M. K. Malhotra, and J. B. Jensen. An evaluation of capacity sensitive order review and release procedures in job shops. *Decision Sciences*, 24(6):1109–1134, 1993.

G. L. Ragatz and V. A. Mabert. An evaluation of order release mechanisms in a job-shop environment. *Decision Sciences*, 19(1):167–189, 1988.

R. Ramasesh. Dynamic job shop scheduling: a survey of simulation research. *Omega*, 18(1):43–57, 1990.

I. Sabuncuoglu and H. Karapinar. A load-based and due-date-oriented approach to order review/release in job shops. *Decision Sciences*, 31(2):413–447, 2000.

R. G. Sargent. An assessment procedure and a set of criteria for use in the evaluation of computerized models and computer-based modelling tools. Technical report, Syracuse University New York Department of Industrial Engineering and Operations Research, 1981.

R. Suri. *Quick response manufacturing: a companywide approach to reducing lead times*. Productivity Press, 1998.

J. Sztrik. Basic queueing theory. *University of Debrecen, Faculty of Informatics*, 193:60–67, 2012.

M. Thürer, C. Silva, and M. Stevenson. Optimising workload norms: the influence of shop floor characteristics on setting workload norms for the workload control concept. *International Journal of Production Research*, 49(4):1151–1171, 2011.

M. Thürer, M. Stevenson, C. Silva, M. J. Land, and L. D. Fredendall. Workload control and order release: A lean solution for make-to-order companies. *Production and Operations Management*, 21(5):939–953, 2012.

M. Thürer, M. J. Land, M. Stevenson, L. D. Fredendall, and M. Godinho Filho. Concerning workload control and order release: The pre-shop pool sequencing decision. *Production and Operations Management*, 24 (7):1179–1192, 2015.

M. Thürer, M. Stevenson, C. Silva, and T. Qu. Drum-buffer-rope and workload control in high-variety flow and job shops with bottlenecks: An assessment by simulation. *International Journal of Production Economics*, 188:116–127, 2017.

H. P. G. Van Ooijen. *Load-based work-order release and its effectiveness on delivery performance improvement*. Citeseer, 1996.

J. K. Weeks. *A simulation study of due-date assignment policies in a dual-constrained job shop*. PhD thesis, University of South Carolina, Business Administration, 1974.

W. Whitt. A review of l aw and extensions. *Queueing Systems: Theory and Appl*, 9, 1992.

R. M. Yerkes and J. D. Dodson. The relation of strength of stimulus to rapidity of habit-formation. *Journal of comparative neurology and psychology*, 18(5):459–482, 1908.

S. Zwartelé. Planning and control in a complex high-mix-low-volume job shop manufacturing environment. Master's thesis, Eindhoven University of Technology, The address of the publisher, 1 2016.

A Overview of sequence and release rules

A.1 Overview of notation

j	job index
m	machine index
o	operation index
t	time index
t_0	current time
o_j	current operation of job j
$o_{j,i}$	the i-th next operation of job j
m_o	machine of operation o
q_m	queue of machine m
l_m	current load of machine m
L_m	load threshold of machine m
d_j	due date of job j
$d_{j,o}$	due date of job j on operation o
a_j	arrival date of job j
$a_{j,o}$	arrival date of job j at operation o
r_j	release date of job j
$p_{j,o}$	processing time of the o-th operation of job j
O_j	number of remaining operations of job j
Q_m	number of jobs in the queue of machine m
J_{shop}	number of jobs on the shop floor
J_{psp}	number of jobs in the pre-shop pool
U_{shop}	shop utilisation at time
$R_{U,V}$	a random variable, drawn from a uniform distribution between U and V
F_i	weighting factors and constants

A.2 Overview of sequence rules

Table A.1: Overview of sequence rules

Name	Abbreviation	Priority measure
First Come First Serve	FCFS	a_j
Last Come First Serve	LCFS	a_j
Earliest Due Date	EDD	d_j
Latest Due Date	LDD	d_j
Operational Due Date	ODD	$d_{j,o}$
Modified Operational Due Date	MOD	$max(d_{j,o}, t_0 + p_{j,o})$
Maximum Job Tardiness	MJT	$max(t_0 - d_j, 0)$
Shortest Processing Time	SPT	$p_{j,o}$
Longest Processing Time	LPT	$p_{j,o}$
Maximum Work Remaining	MWR	$\sum p_{j,o}$
Least Work Remaining	LWR	$\sum p_{j,o}$
Maximum Number of Operations Remaining	MOR	O_j
Lowest Number of Operations Remaining	LOR	O_j
Least Slack Time	LST	$d_j - t_0 - \sum p_{j,o}$
Least Operational Slack Time	LOS	$d_{j,o} - t_0 - p_{j,o}$
Least Slack per Remaining Operation	SPO	$\frac{d_{j,o}-t_0-p_{j,o}}{O_j}$
Least Critical Ratio	LCR	$\frac{d_j-t_0}{\sum p_{j,o}}$
Least Operational Critical Ratio	OCR	$\frac{d_{j,o}-t_0}{p_{j,o}}$
Least Number In Next Queue	LNINQ	Q_x, where $x = m_{o_{j,1}}$
Least Work In Next Queue	LWINQ	$\sum_{j=1}^{Q_x} p_{j,o}$, where $x = m_{o_{j,1}}$
Minimum Slack Ratio	MSR	$\frac{1}{O_j} \sum_{o=1}^{O_j} \frac{p_{j,o}}{T_{m_o}-l_{m_o}}$

A.3 Overview of release rules and their classification

Table A.2: Release rules and their classification

		BIL	MIL	MIL	PIOC	DLF	BFL	FFL	PBB
Order release mechanism	Load limited			✓	✓		✓	✓	✓
	Time phased	✓	✓		✓				
Timing convention	Continuous		✓		✓				
	Discrete	✓		✓		✓	✓	✓	✓
Workload measure	Number of jobs		✓	✓	✓	✓			
	Work quantity	✓					✓	✓	✓
Aggregation of workload measure	Total shop load			✓					
	Bottleneck load								
	Load by each work-centre	✓	✓		✓	✓	✓	✓	✓
Workload accounting overtime	Atemporal	✓		✓	✓	✓			✓
	Time bucketing						✓	✓	
	Probabilistic								
Workload control	Upper bound only			✓			✓	✓	✓
	Lower bound only				✓	✓			
	Upper and lower bounds								
	Workload balancing								
Capacity planning	Active				✓	✓			
	Passive	✓	✓	✓			✓	✓	✓
Schedule visibility	Limited	✓		✓	✓	✓	✓	✓	✓
	Extended								

B Conceptual model

B.1 Conceptual model systematic representation

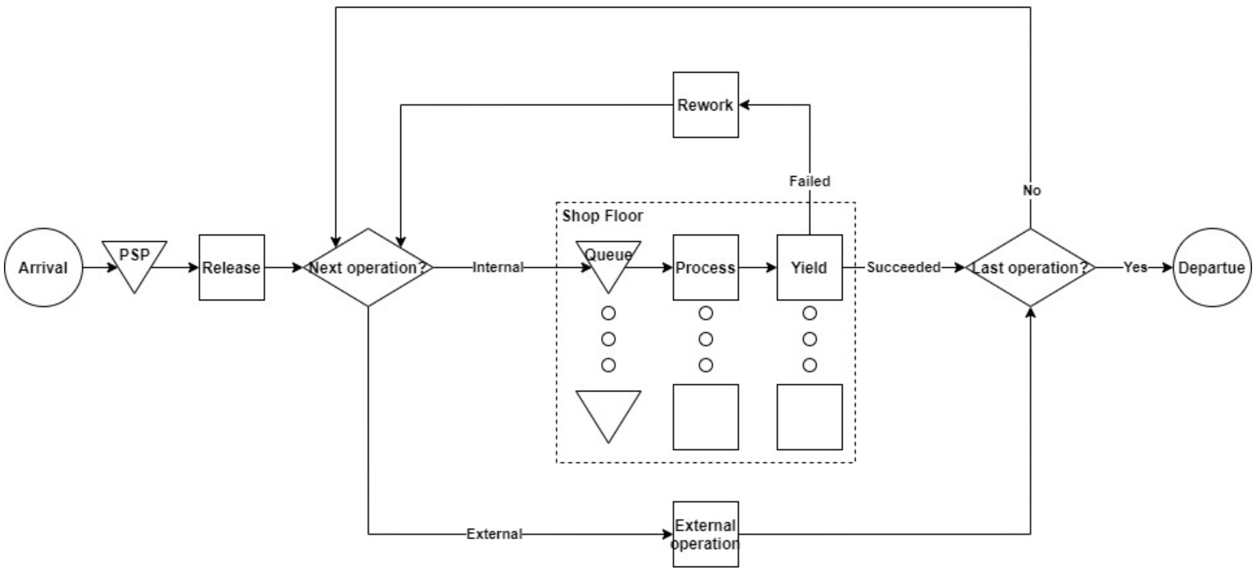

Figure B.1: Conceptual Job Shop Model

B.2 Conceptual model assumptions

1. Material is always available
2. An order is never cancelled
3. An operation cannot be preempted, so once processing begins it cannot be stopped until complete
4. There are no flexible routings
5. Each job can be processed on one and only one machine at a time
6. Setup times are non-sequence dependent
7. Re-entrant jobs are permitted, that is two or more operations of a job may be processed on the same machine
8. Each machine can only be process one job at a time
9. External operations have infinite capacity
10. A machine only breaks-down after finishing a job
11. Each job can only be processed on one machine at a time
12. The shop is open continuously

C Overview of events

Table C.1: Overview of events

Event	Case	State Change		Events Scheduled		
		Object	*State*	*Type*	*Object*	*Time*
Job Arrival		Job	psp	Job Arrival		$t_{now} + t_{iat}$
Job Release	Release			Next Operation		t_{now}
				Job Release		t_{now}
	No release					
Next Operation	Operation internal	Job	que	Check Overtime Policy	Machine	t_{now}
	Operation external			External	Job	t_{now}
	Operation none	Job	dep			
Check Overtime Policy				Start Operation	Machine	t_{now}
Start Processing	Idle machine and empty queue	Machine	pro	End Processing	Machine, Job	$t_{now} + t_{process}$
		Job	pro			
	No queue					
End Processing	Rework			Next Operation	Regular Job	t_{now}
				Rework	Rework Job	t_{now}
	No rework			Next Operation	Job	t_{now}
	Goes down	Machine	dwn	Repaired	Machine	$t_{now} + t_{down}$
	Stays up	Machine	idl	Check Overtime Policy	Machine	t_{now}
Rework				Next Operation	Job	t_{now}
External		Job	pro	Next Operation	Job	$t_{now} + t_{external}$
Repaired		Machine	idl	Check Overtime Policy	Machine	t_{now}

D Data sets

D.1 Data collection

Data on job arrival times, routings, priorities, due dates, waiting times, and process times is collected from a few sources. Each of the datasets obtained from the ERP system BaaN were retrieved using SQL in combination with the iQBS system of VDL ETG Eindhoven.

The first data set, containing the machine routings of all jobs in the shop for the years 2017, 2018, and 2019, and the second data set, containing data on job arrival times and due dates, were obtained from BaaN on January 4'th 2020. Data sets of the priorities of jobs at Parts were also obtained from BaaN, these data sets were retrieved on nineteen days in February, March and April 2020. The last data set which was obtained from BaaN, contained data on process times and waiting times for finished operations and was also retrieved on January 4'th 2020. An overview of these data sets and the data on which the data was retrieved is tabulated in Table D.1.

Table D.1: Overview of data sets

Data set	Information	Retrieved on
Routing	Planned routings	January 4'th 2020
Arrival and due	Arrival dates, requested and confirmed due-dates	January 4'th 2020
Priorities	Current distribution of priorities in the shop	19 days in quarter 1 2020
Finished operations	Start and end dates of finished operations	January 4'th 2020
External operations	Start and end dates of external operations	January 4'th 2020

Since no data was available on the development of the WIP over time the Finished operations data set was used to deduce information on this. This was done by counting the total number of job for which the start date of the first operation was before and the end date of the last operation was after each date in 2017, 2018 and 2019.

D.2 Data quality

In this subsection the quality of part of the *Finished operations* data set is discussed. The relevant data discussed has the structure as shown in Table D.2. *SFCorder* is the id number of an order, *volgnummer* is the id of an specific step in the production of an order, *taak* is an task number combined with an description of the task, *voorgaande gereed* is the date on which the previous task was finished or the order entered the system in case of the first step, and *bewerking gereed* is the date on which the current task was finished.

Here we checked whether the data for each order adheres to three forms of internal consistency. Horizontal consistency, meaning that for each *volgnummer voorgaande gereed* is not larger than *bewerking gereed*, vertical consistency i.e. are *bewerking gereed* within a *SFCorder* for subsequent *volgnummer* non-decreasing, And diagonal consistency is within a *SFCorder* the *bewerking gereed* equal to the next *voorgaande bewerking*.

Table D.2: Example data finished operations

SFCorder	Volgnummer	Taak	Voorgaande gereed	Bewerking gereed
123456	10	1234 - do something	01-01-2019	01-02-2019
	20	5678 - do something else	01-02-2019	01-02-2019
	30	9123 - do even more	01-02-2019	01-04-2019

First check is whether the finish date of the previous operation is smaller than or equal to the finish date of the operation at hand. This is the case for operations in 11 out of orders. Of all the cases this is not true except 1 it is regarding "non-value adding operations" (e.g. "Controle Parts" or "Pluskosten Mechanisch"). Considering the small percentage of faulty values deleting the orders whit these operations should not skew the data.

Secondly it is checked whether following operations have non-decreasing finish times. There is 1 order in which subsequent operations have decreasing finish times. This again is such a small number the deletion of this should not matter.

Thirdly it is checked whether the operation finished date of an operation is equal to the previous operation finished of subsequent operations. This is not the case for 23348 out of 256346 operations or 9.11%. These data irregularities seem to occur in a number of situations. Firstly we see that in 10387 out of the 51924 faulty operations the difference occurs due to external operations. Moreover taking a closer look at the involved operations we see that for the ending operation over 80% of the faults concerns non-value adding operations (e.g. control or blocked) the same is found for starting operations. Further analysis showed that diagonal inconsistency occurs when the same operation spans multiple “Volgnummers”. This is the case for 6863 operations out of the 23348 faulty operations. Further analyses revealed that this happens for one of two reasons. First it happens because batches are split up to prevent idleness. Secondly external operations are not included in this data set and are always followed by one of these ”non-value adding” tasks.

D.3 Data inclusion rules

Since we are selecting three years of data from all the data available, the cut off point of this data had to be determined. There are four ways (inclusion rules) of determining which data points are part of the three year sample and which are not. Rule 1 includes all orders that started before the end of the scope and are not finished before the start of the scope, rule 2 includes all orders that were finished during the scope, rule 3 includes all orders that were started during the scope, and rule 4 includes all orders that were started and finished during the scope. Figure D.1 illustrates these four rules systematically. In case the inclusion of data is cut off at either end of there is a change the remaining jobs will be biased towards jobs, with shorter throughput times and if the data is not cut off there is a change the data is biased towards long jobs. Therefore, the empirical probability density functions for the routing length of jobs for each of the four inclusion rules were compared. From this comparison no clear bias was discovered. Since, it is preferred to analyse jobs which have been finished we decided to use inclusion rule 2 to select the data.

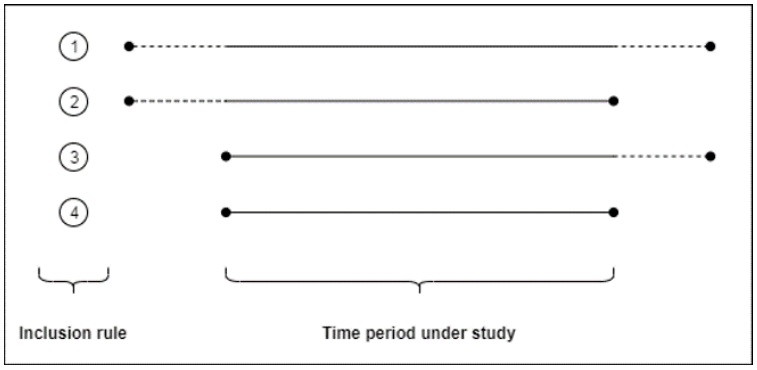

Figure D.1: Data inclusion rules

D.4 Data consolidation

Within the information system of *Parts* operations are called tasks and tasks are associated with machines. Most of these machines are physical machines, however some are just concepts within the information system. These machines are divided into 42 distinct departments, also called work stations. Moreover tasks on machines can be executed manned, partly manned, or unmanned. Within this study the decision has been made to consolidate all different types of task belong to a machine to that single machine, and further consolidating all machines into the 42 work stations. As a result, each time we refer to a machine from the data in this thesis (with the exclusion of this appendix), we actually refer to a workstation.

E Overview of simulation parameters

E.1 Distributions used

Table E.1: Overview of distributions used

Distribution	Type
Inter arrival time	Exponential distribution
Job size	Discrete uniform distribution
Job priority	Empirical distribution
Routing	Empirical distribution
Job due date	Empirical distribution
Process time	Log normal distribution
Machine breakdown	Bernoulli distribution
Down time	Exponential distribution
Machine yield	Binomial distribution
External time	Log normal distribution

E.2 Overall parameters

Table E.2: Overview of overall simulation parameters

Parameter	Value
Arrival rate	2.620
Maximum job size	5.000
Overtime rate	0.909
Break down probability	0.100
Machine yield	0.985
External processing time mean	87.470
External processing time variation	297.960

E.3 Priority parameters

Table E.3: Distribution of job priorities

Priority Level	Fraction of jobs	Corresponding interval
1	68.1%	(0.000, 0.681)
2	3.2%	(0.681, 0.731)
3	5.6%	(0.731, 0.769)
4	13.3%	(0.769, 0.902)
5	9.8%	(0.902, 1.000)

E.4 Due date parameters

Table E.4: Distribution of job priorities

Due date tightness	Fraction of jobs	Corresponding interval
0.0	0.2%	(0.000 , 0.002)
0.1	0.6%	(0.002 , 0.008)
0.2	0.8%	(0.008 , 0.016)
0.3	1.0%	(0.016 , 0.026)
0.4	1.5%	(0.026 , 0.041)
0.5	2.2%	(0.041 , 0.063)
0.6	2.9%	(0.063 , 0.091)
0.7	3.7%	(0.091 , 0.129)
0.8	4.3%	(0.129 , 0.172)
0.9	5.0%	(0.172 , 0.222)
1.0	5.3%	(0.222 , 0.275)
1.1	6.0%	(0.275 , 0.336)
1.2	6.8%	(0.336 , 0.405)
1.3	7.4%	(0.405 , 0.479)
1.4	8.1%	(0.479 , 0.560)
1.5	7.9%	(0.560 , 0.640)
1.6	6.6%	(0.640 , 0.707)
1.7	4.9%	(0.707 , 0.756)
1.8	3.7%	(0.756 , 0.794)
1.9	2.9%	(0.794 , 0.823)
2.0	2.5%	(0.823 , 0.848)
2.1	1.9%	(0.848 , 0.868)
2.2	1.5%	(0.868 , 0.883)
2.3	1.5%	(0.883 , 0.898)
2.4	1.3%	(0.898 , 0.911)
2.5	1.1%	(0.911 , 0.921)
2.6	1.0%	(0.921 , 0.931)
2.7	0.8%	(0.931 , 0.939)
2.8	0.7%	(0.939 , 0.947)
2.9	0.7%	(0.947 , 0.953)
3.0	0.6%	(0.953 , 0.959)
3.1	0.5%	(0.959 , 0.964)
3.2	0.4%	(0.964 , 0.968)
3.3	0.4%	(0.968 , 0.971)
3.4	0.3%	(0.971 , 0.975)
3.5	0.3%	(0.975 , 0.978)
3.6	0.3%	(0.978 , 0.981)
3.7	0.3%	(0.981 , 0.984)
3.8	0.3%	(0.984 , 0.987)
3.9	0.3%	(0.987 , 0.990)
4.0	0.2%	(0.990 , 0.992)
4.1	0.2%	(0.992 , 0.995)
4.2	0.2%	(0.995 , 0.996)
4.3	0.1%	(0.996 , 0.998)
4.4	0.1%	(0.998 , 0.998)
4.5	0.2%	(0.998 , 1.000)

E.5 Machine parameters

Table E.5: Machine process times

Machine	Process Time	
	Mean	Variation
1	1.94	3.98
2	83.21	61.83
3	5.63	4.53
4	10.83	10.61
5	48.98	21.69
6	2.60	6.34
7	3.19	3.97
8	7.31	11.49
9	1.34	3.28
10	7.47	7.67
11	1.70	6.28
12	0.93	3.29
13	5.24	5.58
14	4.00	6.52
15	4.18	10.08
16	0.24	1.02
17	3.66	4.33
18	12.73	18.98
19	2.41	6.54
20	1.79	8.46
21	1.28	5.48
22	1.49	9.39
23	2.73	6.84
24	9.29	13.94
25	10.15	14.36
26	4.03	6.92
27	5.07	9.52
28	10.37	10.98
29	21.43	46.92
30	3.04	5.54
31	7.06	13.17
32	0.43	1.46
33	0.15	1.27
34	1.45	4.11
35	4.29	2.99
36	0.32	1.72
37	0.20	0.62
38	1.12	8.17
39	1.53	4.71
40	0.50	1.72
41	1.24	3.79
42	87.47	297.96

F Validation and verification tables and overviews

F.1 Overview of face validation relations

1. As arrival rate goes up work in progress goes up and to infinity as utilization goes to 1
2. As arrival rate goes up waiting time goes up and to infinity as utilization goes to 1
3. As arrival rate goes up delivery performance goes down and to 0 as utilization goes to 1
4. As service rate goes down work in progress goes up and to infinity as utilization goes to 1
5. As service rate goes down waiting time goes up and to infinity as utilization goes to 1
6. As service rate goes down delivery performance goes down and to 0 as utilization goes to 1
7. As yield goes down work in progress goes up and to infinity as utilization goes to 1
8. As yield goes down waiting time goes up and to infinity as utilization goes to 1
9. As yield goes down delivery performance goes down and to 0 as utilization goes to 1
10. As downtime goes up work in progress goes up and to infinity as utilization goes to 1
11. As downtime goes up waiting time goes up and to infinity as utilization goes to 1
12. As downtime goes up delivery performance goes down and to 0 as utilization goes to 1
13. As breakdown frequency goes up work in progress goes up and to infinity as utilization goes to 1
14. As breakdown frequency goes up waiting time goes up and to infinity as utilization goes to 1
15. As breakdown frequency goes up delivery performance goes down and to 0 as utilization goes to 1
16. As routing length goes up work in progress goes up and to infinity as utilization goes to 1
17. As routing length goes up waiting time goes up and to infinity as utilization goes to 1
18. As routing length goes up delivery performance goes down and to 0 as utilization goes to 1
19. As variations increase work in progress goes up
20. As variations increase waiting time goes up
21. As variations increase delivery performance goes down
22. As due date tightness goes up delivery performance goes down

F.2 Comparison to queuing models

Table F.1 illustrates the mathematical and simulation results for a m/m/1 queue with parameters $\mu = 5$ and $\lambda = 3$. Table F.2 illustrates the mathematical and simulation results for a m/m/1 queue with priority and parameters $\mu = 3$, $\lambda_1 = 1.5$, and $\lambda_2 = 1.5$. Jobs of type 1 have priority over jobs with type 2.

Table F.1: Comparison to m/m/1 queue

	Work in progress	Waiting time	Throughput time
Calculated	1.5	0.3	0.5
Simulated	1.492	0.298	0.498

Mathematical and simulation results for a m/m/1 queue with parameters $\mu = 5$ and $\lambda = 3$.

Table F.2: Comparison to m/m/1 priority queue

		Work in progress	Throughput time
Type 1	Calculated	0.557	0.371
	Simulated	0.558	0.372
Type 2	Calculated	0.943	0.629
	Simulated	0.945	0.630

Mathematical and simulation results for a m/m/1 queue with priority and parameters $\mu = 3$, $\lambda_1 = 1.5$, and $\lambda_2 = 1.5$ where jobs of type 1 have priority over jobs with type 2.

F.3 Comparison to Jackson Networks

Table F.3 illustrates the mathematical and simulation results for a Jackson Network with the structure as illustrated in Figure F.1, parameters $\lambda = 3$, and $\mu_A = \mu_B = \mu_C = \mu_D = 5$ and transition rates $p_{A,A} = \frac{1}{9}$, $p_{A,B} = \frac{6}{9}$, $p_{A,C} = \frac{1}{9}$, $p_{B,C} = \frac{1}{9}$, $p_{B,D} = \frac{8}{9}$, $p_{C,B} = \frac{1}{9}$, and $p_{C,D} = \frac{8}{9}$.

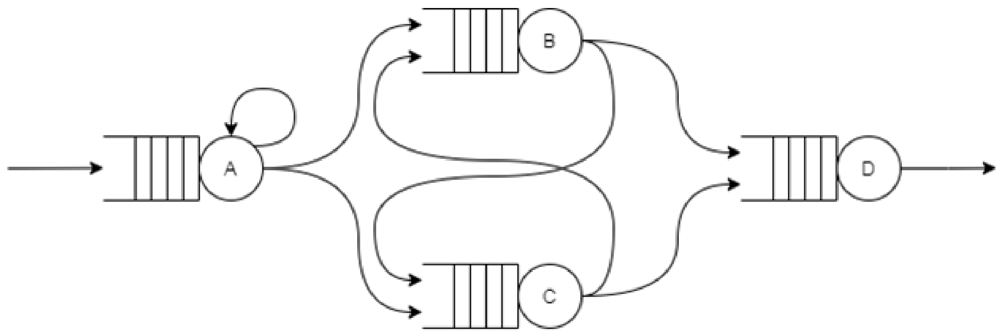

Figure F.1: Structure of Jackson Network for comparison

Table F.3: Comparison to Jackson Network

	Work in progress	Waiting time	Utilisation			
			A	B	C	D
Calculated	4.727	1.546	0.675	0.203	0.476	0.600
Simulated	4.666	1.556	0.674	0.202	0.472	0.599

Mathematical and simulation results for a Jackson Network with parameters $\lambda = 3$, and $\mu_A = \mu_B = \mu_C = \mu_D = 5$ and transition rates $p_{A,A} = \frac{1}{9}$, $p_{A,B} = \frac{6}{9}$, $p_{A,C} = \frac{1}{9}$, $p_{B,C} = \frac{1}{9}$, $p_{B,D} = \frac{8}{9}$, $p_{C,B} = \frac{1}{9}$, $p_{C,D} = \frac{8}{9}$, and where jobs leave after operation D.

F.4 Comparison to yield formulas

Table F.4 illustrate the expected number of operations needed before finishing a job for different yield percentages and an initial routing length equal to ten where a job is scrapped upon failure or a job is repaired once before scrapping on the second failure.

Table F.4: Comparison to extended yield formula

		Yield			
		1.00	0.95	0.90	0.85
Simple yield	Calculated	10.00	13.40	18.83	27.53
	Simulated	10.00	13.38	18.83	27.53
Extended yield	Calculated	10.00	11.20	13.68	17.76
	Simulated	10.00	11.21	13.75	17.64

Expected number of operations needed before finishing a job for different yield percentages and rework types.

G Model sensitivity

G.1 Model sensitivity with respect to work in progress

Table G.1: Model sensitivity with respect to work in progress

Parameter changed	Increase	Decrease
Arrival rate	-10.9%	10.0%
Processing time mean	-10.6%	17.2%
Processing time variation	-6.0%	3.0%
External operation time mean	-20.9%	15.9%
External operation time variation	-1.8%	2.1%
Machine downtime mean	-0.6%	2.2%
Machine downtime frequency	-7.9%	1.6%
Yield	2.6%	-20.0%

G.2 Model sensitivity with respect to effective yield

Table G.2: Model sensitivity with respect to effective throughput

Parameter changed	Increase	Decrease
Arrival rate	0.1%	-0.3%
Processing time mean	-0.1%	0.9%
Processing time variation	-0.1%	0.6%
External operation time mean	-0.9%	1.0%
External operation time variation	-0.1%	0.7%
Machine downtime mean	0.0%	0.2%
Machine downtime frequency	0.0%	0.3%
Yield	0.1%	-0.1%