Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

TECHNISCHE UNIVERSITÄT WIEN

DIPLOMARBEIT

Solution approaches to a flexible job-shop scheduling problem with a quality control as stochastic element

ausgeführt am

Institut für
Stochastik und Wirtschaftsmathematik
TU Wien

unter der Anleitung von

Senior Lecturer Dipl.-Ing. Dr.techn.
Josef Leopold Haunschmied

and

Privatdoz. Mag.rer.soc.oec. Dr.rer.soc.oec.
Raimund Kovacevic

durch

Valentina Maria Einspieler

Matrikelnummer: 01426372

Wien, am 24. März 2020

Unterschrift Betreuer

Kurzfassung

Diese Arbeit behandelt ein reales Produktionsplanungsproblem mit parallelen Maschinen. Probleme dieser Art zeichnen sich dadurch aus, dass jeder Auftrag eine unterschiedliche Zusammensetzung sowie Reihenfolge von Operationen absolvieren muss und für jede dieser Operationen eine oder mehr Maschinen zur Verfügung stehen. Der Prozess, den wir betrachten, unterscheidet sich von anderen insbesondere durch eine Qualitätskontrolle. Diese ist ein für alle Aufträge verpflichtender Produktionsschritt, den diese jeweils so oft durchlaufen müssen bis sie freigegeben werden. Jeder Auftrag wird jedoch lediglich mit einer bestimmten Wahrscheinlichkeit akzeptiert, weshalb die Qualitätskontrolle ein stochastisches Element des Prozesses darstellt. Da jede negative Beurteilung durch die Kontrolle bedeutet, dass der jeweilige Auftrag nachgearbeitet werden muss, hat der Ausgang jeder Qualitätskontrolle einen großen Einfluss auf den gesamten Produktionsplan.

Das Ziel dieser Arbeit ist, einen optimalen Produktionsplan unter Berücksichtigung der Qualitätskontrolle als stochastische Komponente des Prozesses zu erstellen. Dazu werden verschiedene Lösungsansätze mithilfe exakter Methoden entwickelt. Zunächst erarbeiten wir ein deterministisches MILP Modell, das den Prozess abbildet. Darauf basierend wird ein robustes sowie ein stochastisches Modell präsentiert. Des weiteren diskutieren wir einen prozessbegleitenden Ansatz, bei dem noch während der laufenden Produktion auf Basis der Ergebnisse der Qualitätskontrollen umgeplant werden kann.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Abstract

In this thesis, we consider a real world flexible job-shop scheduling problem. This kind of problem is a situation whereby each job has to follow an individual route through the process and for each operation more than one machine might be eligible. The special property of the process at hand that sets it apart from other problems of this class is a mandatory quality control step as part of the process. Every job has to repeat this step until it is approved. However, each job will pass the control only with a certain probability and, thus, this part represents a stochastic element of the process. Since a rejection by the quality control entails a rework procedure for the respective job, the outcome of each quality control will have a huge effect on the whole schedule.

The purpose of this thesis is to obtain an optimal schedule considering the stochastic characteristic of the quality control. In order to do so, we discuss various solution approaches of exact methods. First, a deterministic MILP formulation of the problem at hand is developed. Based on this model, we apply robust optimization techniques to the problem as well as develop a stochastic counterpart to the deterministic model. Additionally, a reactive rescheduling approach is presented. In contrast to the previous approaches, this provides the possibility to perform adjustments to the schedule based on the outcomes of the quality controls after the process has already started.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Acknowledgement

First of all, I want to thank the company *MCP Algorithm Factory* for the incredible experiences during my internship. I am especially grateful to Dipl.-Ing. Christoph Mrkwicka who helped me find the exciting topic for this thesis and always made time to support me in any way possible.

I would like to express my sincere gratitude to my supervisor Senior Lecturer Dipl.-Ing. Dr.techn. Josef Haunschmied. By continuously providing me with feedback as well as support on how to proceed, he made a perpetual workflow and an enjoyable working environment possible. Also, I am thankful to my second advisor Privatdoz. Mag.rer.soc.oec. Dr.rer.soc.oec. Raimund Kovacevic, who reviewed the content of my work and provided ideas for enhancements.

Last but by all means not least, I want to thank my family, especially my parents. They provided me with support of any kind every step of the way up to this point and encouraged me to always give everything no matter what the task is.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am 24. März 2020

Valentina Maria Einspieler

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Contents

List of Figures **III**

1 Introduction **1**

2 Problem description **6**

2.1 Important definitions 6
2.2 Process structure 8
2.2.1 Constraints and bottlenecks 10
2.2.2 Objective of scheduling 11
2.2.3 Characteristics of the process industry 12
2.3 Problem classification 13
2.4 The quality control as stochastic element 14

3 Model formulation of the problem **15**

3.1 General solution requirements 15
3.1.1 Feasibility 15
3.1.2 Special solution requirements and properties 16
3.1.3 Assumptions 17
3.2 Mixed Integer Linear Programming formulation 17
3.2.1 Parameters 18
3.2.2 Variables 20
3.2.3 Determinstic model 20
3.2.4 Complexity 25
3.2.5 Implementation in MATLAB 26
3.2.6 Computational limits 32
3.2.7 Applicability to problems of real world dimensions 32

4 Robust approach **34**

4.1 Motivation 34
4.1.1 Deterministic solution 35
4.1.2 Impact of quality control's outcome on production schedule 36

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

4.2 Representing uncertainty in the model . . . 37
4.3 Robust scheduling . . . 38
4.4 Metrics for robustness . . . 39
4.5 Robust model 1 . . . 39
4.5.1 Application to the problem . . . 41
4.5.2 Remarks to the approach . . . 42
4.6 Transformation of deterministic model . . . 43
4.7 Robust model 2 . . . 46
4.7.1 Application to the problem . . . 47
4.7.2 Implementation in MATLAB . . . 50
4.7.3 Remarks and summary of the approach . . . 53

5 Stochastic Model **54**
5.1 Introduction to stochastic programming . . . 56
5.1.1 Objective function depending on random variables . . . 56
5.1.2 Chance constraint stochastic optimization . . . 57
5.1.3 Extensions . . . 58
5.2 Stochastic model . . . 58
5.2.1 Stochastic approach . . . 58

6 Reactive scheduling **67**
6.1 Model predictive approach . . . 67
6.2 Rescheduling algorithm . . . 68
6.3 Extensions . . . 74
6.4 Problems of this approach . . . 75

7 Conclusion **76**

Bibliography **78**

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

List of Figures

2.1 Depiction of the process structure, own drawing 9

3.1 The optimal schedule for **Example 3.1**, own drawing 31

4.1 The optimal schedule for **Example 4.1**, own drawing 35
4.2 The realization of the schedule for **Example 4.1** in case job 1 gets rejected once, own drawing 36
4.3 The robust schedule for **Example 4.2** with $\Gamma = 1$, own drawing 52

5.1 The realization of the schedule for **Example 4.1** in case job 3 gets rejected once, own drawing 54
5.2 The optimal schedule with perfect foresight for **Example 4.1** in case job 3 gets rejected once, own drawing 55
5.3 Scenario Tree for **Example 5.1**, own drawing 62

6.1 The optimal schedule in the worst case scenario of two tries per job, own drawing 70
6.2 Scenario tree for the worst case schedule of **Example 6.1**, own drawing . . 71
6.3 The schedule until the next quality control in case job 3 is approved, own drawing 72
6.4 The actual realization of the process plan for **Example 6.2**, own drawing . 74

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

1 Introduction

There exist various types and applications of scheduling problems. Those include not only optimal timetables for buses, schools and tournaments but also medical and production scheduling problems. They are found especially in applications in the research areas of engineering, operations research and computer science (Pinto and Grossmann (1998)).

The scheduling in the field of production can be defined as a "decision making process to determine when, where and how to produce a set of products given requirements in a specific time horizon, a set of limited resources, and processing recipes" (Floudas and Lin (2004)). As emphasized in Maravelias (2012), this allocation is not always restricted to the question of how to allocate resources over time but might in some cases also entail how the tasks should be assigned to resources and how to sequence those.

In Méndez et al. (2006) a classification of batch processes is provided, which includes 13 aspects with various possible characteristics and typical features of such processes regarding the general process layout, changeovers as well as the degree of uncertainty. This already gives a hint on how diverse the problems can be and how many different factors have to be taken into account which influence the structure and the complexity of a production scheduling problem (Méndez et al. (2006)).

Unlike other optimization problems, for which the aim of the optimization might seem quite obvious, when dealing with scheduling problems, the objective(s) might be unambiguous or even conflictive (Kallrath (2002)). The focus of the optimization in a production scheduling problem might for instance either be to minimize the total make span in general or - comparably - to maximize the throughput. On the other hand, regarding customer satisfaction and reliable delivery, an additional objective might be to maximize the number of orders that are completed at a prescribed due date (Li and Ierapetritou (2008)). Moreover, one might also aim at reducing costs for e.g. logistics as well as costs that arise due to an inefficient production. In Zhang and Yan (2005), for instance, the objective includes inventory costs during the manufacturing process as well as penalty costs for over- and underproduced products. They also include setup costs, when machines have to be adjusted for different types of products. All of these objective functions minimizing certain costs might include actual monetary costs but can also reflect costs in the sense of time loss or material waste.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

The production scheduling problem which is discussed in this thesis is one that arises in the chemical industry and at the end of the production line, the main output are types of silicone. The chemical industry forms part of the process industry, which has to be distinguished from the assembly industry. In the assembly industry the production mainly consists of making and putting together components, which is why it is also called discrete industry (Jovan (2002)). Examples are car or furniture industries. The main focus in such industries lies on managing work queues at machines and making sure that all components are ready when they are needed for the next operation (Jovan (2002)). In the process industry, on the other hand, which is also called continuous industry, usually products are blended together in large quantities. In this case no discrete units can be identified. Typical examples of the process industry are the chemical industry, oil refineries, pulp and paper industries and pharmaceutical industries. Their production processes often contain physical or chemical reactions. In the process industry usually mass production and repetitive manufacturing is found, i.e. a low variety of products but in a high volume is manufactured (Jovan (2002)). Nevertheless, some batch manufacturing factories exist within the process industry as well, which produce an intermediate number of products in an intermediate quantity (Jovan (2002)). This might be found for instance in the production of pharmaceutical products.

A major distinction regarding the production environment in the process industry has to be made between *job-shop* and *flow-shop* production (Ierapetritou and Li (2009)). In a flow-shop production all jobs have to complete the same steps in the same order, which means that all jobs follow an identical route (Li and Ierapetritou (2008)). Thus, the sequencing of the jobs is of utmost importance for the scheduling. In a job-shop production, on the other hand, there are different kinds of resources, typically machines, through which each job uses an individual way. Since the route of each job differs from the one of the others, the major goal of scheduling lies in the allocation of jobs to machines over time (Šeda (2007)). The job-shop production is characterized by a high variety in types of output but in low volume and the design of the product is typically customized.

In Jovan (2002) several characteristics and difficulties are discussed that commonly arise when dealing with scheduling problems in the process industry. This includes a complex composition of materials and resources, not having the possibility to store the batches in between operations and high changing times or costs for operating on different product types in turn. We will discuss the presence of these characteristics in the problem at hand in the next chapter once the structure of the process and its restrictions have been introduced.

In addition to the characteristics in Jovan (2002), Honkomp et al. (2000) emphasizes on

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

the problem size. Especially in scheduling problems arising in the process industries the number of final products as well as the possibility of using alternative resources or routes through the production process can result in a very complex scheduling task (Honkomp et al. (2000)).

Another major complexity in problems of the process industry is the presence of one (or even more) sources of uncertainty as described and classified in Li and Ierapetritou (2008). This means that the actual parameters like the availability of raw material, the prices or the demand are not known or subject to unexpected deviations at the time the schedule has to be made but attains its value at a later point in time (Ierapetritou and Li (2009)). Apart from uncertainty coming from the market, also internal uncertainty such as machine reliability is an important factor to take into consideration. In Li and Ierapetritou (2008) the different kinds of uncertainties are classified into the following four categories.

type of uncertainty	example
model inherent	kinetic constants, physical properties
process inherent	flow rates, temperature variations or processing times
external	demand, prices or environmental conditions
discrete	equipment or employee availabilities

Table 1.1: Classification and examples for uncertainties in the process industry, Li and Ierapetritou (2008)

The result we seek in this thesis is a detailed production schedule for the problem at hand providing the time and machine for each operation such that the predefined set of orders is completed in the shortest time possible. A vast number of models as well as solution approaches to production scheduling problems can be found in the literature and many excellent reviews exist. For example, Kallrath (2002) provides an extensive review about the process industry and Maravelias (2012) includes a detailed model and solution approach classification in the chemical industry. In Honkomp et al. (2000) the main difficulties that arise in typical problems occurring in the process industry are highlighted, such as the presence of uncertainty. Ierapetritou and Li (2009) give an extensive review about modeling uncertainty and discuss various solution approaches.

As it will become clear in the detailed description of the problem in the next chapter, the specific part of the process that we consider has very special properties. These properties set the scheduling problem apart from other problems discussed in the literature, which entails that the existing models and solution approaches have to be adapted to the particular requirements.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

Two main special properties of the process can be identified. The first one is the requirement that all operations have to be performed within a vessel, which are only of limited supply. Thus, the number of jobs that are processed at the same time needs to be limited to the number of vessels available. We have accomplished this in the model by incorporating a way of keeping count on jobs entering and leaving the system in order to restrict the number of jobs in the processing at the same time.

The other major characteristic affecting the solution of the problem is the stochastic nature of a quality control step within the process. Every job has to be approved by the testing procedure which only happens with a certain probability. Since the job needs to repeat the steps preceding the quality control before it can be tested again, it is easy to imagine that an unexpected rejection of a job can completely derail the schedule.

We address in this thesis the novel element of a quality control step within the process. Since the scheduling problem stems from a real world production planning problem, it is of importance to discuss approaches on how to model the problem including all its special restrictions that need to be fulfilled in order to obtain a production plan that is able to be implemented in reality. To deal with the quality control as a stochastic element we will interpret it as a scheduling problem with uncertain processing times, for which various literature exists, for instance Balasubramanian and Grossmann (2002), Shafia et al. (2011) or Luh et al. (1999).

In this thesis we will present approaches of exact methods to obtain a schedule for the process at hand. The goal will be twofold. On the one hand, it will be important to produce a solution that accounts for possible deviations from the original schedule due to the random outcome of the quality control in such a way that the schedule stays feasible. On the other hand, we will want to make sure that the objective, which is the time it takes to finish a given set of jobs, is minimized.

The first approach completely neglects the presence of uncertainty in the process and finds the schedule as a solution to a mixed integer linear programming (MILP) model. An overview of such models is provided for instance in Grossmann et al. (1996). The model for the problem at hand is based on the one given in Mousakhani (2013) for a general flexible job-shop problem including setup times. This model is extended by constraints ensuring the vessel restrictions, which are based on the ones presented in Caumond et al. (2009) limiting the number of jobs in the system at the same time. This deterministic model forms the basis for all other approaches presented in the subsequent chapters.

Next, we focus on finding a schedule which stays feasible or does not deteriorate for different realizations of the stochastic variables. These considerations are based on robust optimization techniques. Lin et al. (2004) present a robust counterpart of a MILP model

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

in the case that the uncertain parameters attain values within a bounded set. In order to increase the trade-off between optimality and robustness, Bertsimas and Sim (2004) propose a robust problem formulation in which the level of robustness can be adjusted. Both of the aforementioned approaches are applied to the problem at hand.

The presented stochastic approach uses the theory of stochastic optimization and minimizes the expected objective value of the deterministic model such that the constraints hold for every scenario that is considered. Thus, the stochastic approach also finds a robust solution but incorporates the probabilities of occurrence for each scenario into the optimization. An introduction to stochastic programming is for example given in Ahmed (2017) or in Birge and Louveaux (2011). The stochastic model in this thesis is developed based on the idea presented in Balasubramanian and Grossmann (2002).

The last approach is a reactive scheduling method in which the schedule is constantly monitored and is adjusted depending on the events (quality controls) observed. Rescheduling algorithms highly depend on the problem considered in order to deliver an efficient solution. A detailed summary of the respective literature is given for instance in Ierapetritou and Li (2009). The approach used in this thesis combines the deterministic with the stochastic approach.

The thesis is structured as follows. The next chapter provides a detailed description of the process and the corresponding problem, discussing all constraints that need to be fulfilled. After the problem is presented verbally, it is modeled as a deterministic mixed integer linear programming problem disregarding the stochastic characteristic of the quality control, which is done in chapter 3. In chapter 4, we motivate incorporating the uncertainty into the model and discuss how to obtain a robust schedule. After a short introduction of the ideas behind stochastic programming, the stochastic variant of the deterministic model is developed in chapter 5. Chapter 6 is concerned with the reactive rescheduling approach. The thesis is closed by a short summary of the approaches and ideas for further research directions.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

2 Problem description

This chapter shall provide the reader with an introduction to the main setup of the process on which we focus in this thesis as well as a profound understanding of the resulting difficulties and problems. After defining terms which will be used frequently in the rest of this work, a general overview of the process structure and the inherent constraints is given and the problem is classified. The chapter is completed by a discussion of the consequences the stochastic element within the process has on the scheduling problem.

2.1 Important definitions

Before looking at the detailed description of the scheduling problem, it is vital to state definitions of terminology used throughout this work. This is of importance to avoid lack of clarity especially because the definitions of certain terms might vary among different literature in the chemical production scheduling, as claimed for example in Maravelias (2012).

product (type) A product or product type is a specific kind of end-result that shall be produced in the process. Every product type has its distinct recipe that specifies input materials with their respective quantities as well as the selection and sequence of the machines that have to be used for the processing.

job A job is an order of a product that has to be produced. It belongs to a specific product type and comes with additional individual information on properties or constraints such as processing times or a due date, i.e. the latest point in time at which the order has to be completed. This means that there might be different jobs of the same product type in the process at the same time, which might still differ in certain properties. The term "**batch**" will be used synonymously but to emphasize on the actual physical entity, in contrast to the abstract description of a job.

resource In the process industry "resources" is not to be understood in terms of stock materials but instead it denotes equipment that can be used for the processing or storing

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

of jobs. In the first case, the resource might be a certain kind of machine, e.g. for mixing or filling, whereas in the second case it refers to storage units such as vessels or tanks, in which either a machine process is operated or the batch is put on hold until it is released for further processing.

task A task is a specific operation in the process, which changes the properties, such as the composition or consistency of a batch, in some kind of way. This might for instance be heating or mixing a batch. It may also include cleaning procedures and other maintenance operations. Both terms, "**operation**" and "task", will be used throughout this thesis as synonyms.

stage/step A stage or, synonymously, step of a process is a specific part of a process. It will refer to a particular kind of operations that are carried out by a certain number of resources. Thus, one stage or step of the process might consist of one or more tasks. For example, the stage "mixing" might consist of the tasks adding ingredients to the blending in addition to the actual mixing task.

operation plan This will be used as a synonym to the term "recipe" used above. It is an ordered set of stages or tasks that a job has to complete for the production of a blending of the required product type.

production plan It is a synonym for "**schedule**" and will contain the information of the time intervals in which the operations will be processed and to which machine they are assigned.

processing time The processing time is the total amount of time (in the discrete case given by the number of time periods) that one operation of the process takes, e.g. the time it takes machine i to process an operation l.

throughput The throughput of a process is the number of orders that are completed within a particular timeframe.

make span The make span is the total amount of time it takes until a predefined set of jobs to be processed are completely finished. This entails that they entered the process, followed the way through the process according to their respective operation plan and finally reached the last stage, where they are stored and no further processing is needed.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

setup time This is the total amount of time it takes to prepare a resource before it can be used (again). It might entail cleaning a machine or adjusting it to the next job that is supposed to be processed according to the schedule. For example, in a production process where a machine is responsible for applying color, the machine has to be cleaned before the next operation starts in the case that different color-types succeed each other.

2.2 Process structure

The part of the production process that we want to discuss consists of the following steps, which are also illustrated in figure 2.1. From now on, the term "process" will only refer to the part of the production process that is within the dashed lines of figure 2.1. Every job, regardless of the product type, has to go through all the following steps from top to bottom.

1. For each job the necessary input materials of different kinds have to be gathered. The quantities used for this kind of product are according to the type-specific recipe. These input materials can have two sources.

   - They can come directly from a source or a supplier,
   - or the raw materials coming from the supplier have to be processed beforehand and are then stored in tanks, before they are actually processed.

   Either way, the provision of these input materials is not of our concern and we take these as exogenously given to the process.

   When a job starts its way through the process, the input materials are filled into a vessel, in which all tasks will be performed. This means, that a batch stays in a vessel without interruption as long as it is processed. The vessels, in which the input materials are put together, have to fulfill necessary hygiene requirements to be used for the processing of a kind of product type.

2. After filling the vessels, the compound of ingredients is now mixed by one of the mixing machines. This takes a certain amount of time, depending on the job's product type.

3. Once the mixing is done, i.e. the processing time for this operation has elapsed, the resulting blend has to undergo a quality check, which is done by sending a sample to an external quality control. Depending on the outcome of the quality control, there are two possibilities how the batch continues its way through the process.

   approval After being approved, the blending is ready for the next and last stage which is being bottled into a tank.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

rejection The blending, the way it is now, is rejected and therefore steps 2 and 3 have to be repeated. This means that, if necessary, ingredients are added to the batch and it is mixed again. Then, it undergoes the quality check once more. This is repeated until the batch is finally approved.

4. The last stage is the bottling station. Every blend that has been approved by the quality control is bottled into a tank. The results of this stage are different kinds of blending that will afterwards be used by the succeeding stages of the production chain.

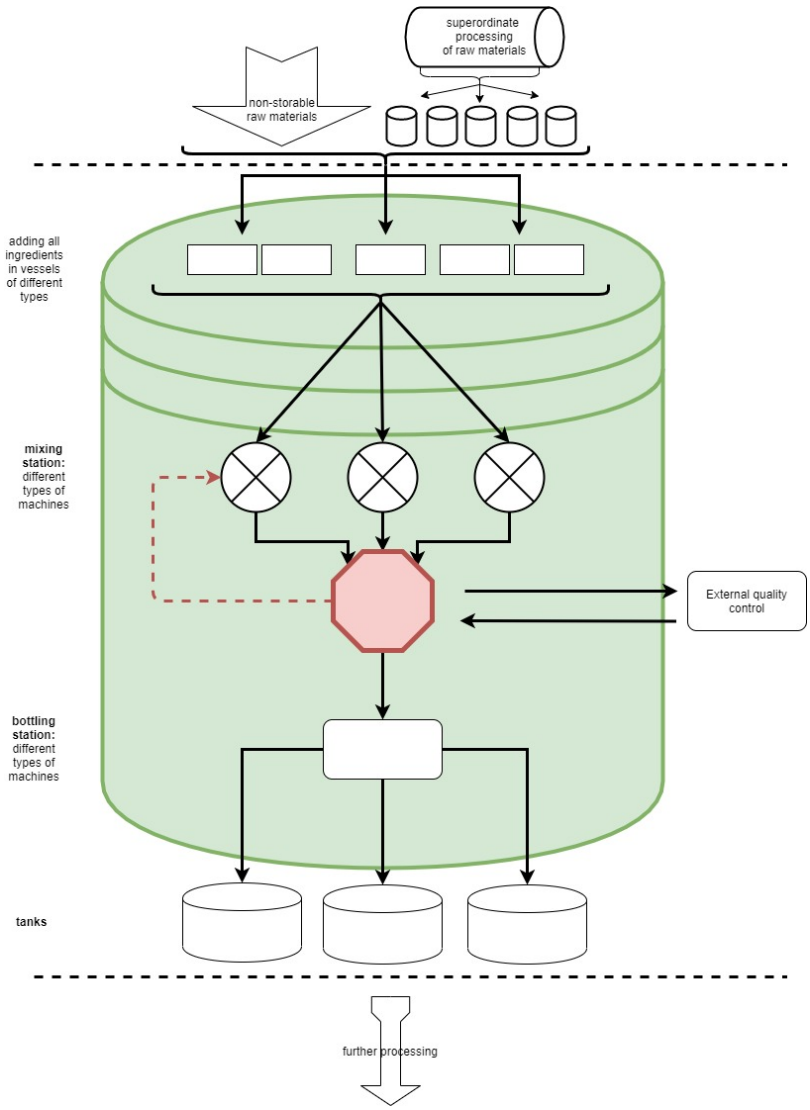

Figure 2.1: Depiction of the process structure, own drawing

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

2.2.1 Constraints and bottlenecks

The general structure described in the previous chapter already gives an idea what kinds of difficulties and constraints are involved in this scheduling problem. Nevertheless, important problem-specific aspects are discussed and highlighted in the following. This will be done verbally and intuitively, whereas later on, in chapter 3, the constraints are given in a formal way.

We start at the top of figure 2.1, at the beginning of every job that goes through the system. It is important to note that the batch size is predetermined and unlike other examples of production processes in the chemical industry (Maravelias (2012)), in this problem the batches cannot be splitted and processed separately nor can batches of different jobs be merged together. Moreover, tasks are not allowed to be preempted, i.e. stopped and proceeded at a later point in time. However, in between the tasks it is permitted that the batches stay in the vessels to wait.

input materials It has already been noted above that the way the input materials are produced or at what time they are provided is not within our scope of influence or decision. Nevertheless, it is vital to recognize the implication of the scheduling of our part on its preliminary stage, the provision of the materials. On the one hand the input materials have to be ready for usage as soon as they are needed but on the other hand must not be delivered too early in case of non-storable materials.

vessels The most important constraints of this process are those regarding the vessels. The main limitation is the number of vessels available because this equals the maximum number of jobs that can be processed at the same time. In addition to this restriction, another constraint arises due to the fact that the vessels might not be uniform. They can vary in volume and other physical properties. This implies that not every vessel is suitable to process any type of product.

Already without studying the process in depth, the availability of vessels can be detected as a major bottleneck. Since all batches need to be filled into vessels for mixing and stay there until they are bottled into the tanks, every vessel is occupied by one batch for virtually the whole time it is in the process. As long as all vessels are occupied, the input materials have to be put on hold. This becomes relevant especially, as mentioned before, when we are dealing with perishable input materials that need to be processed within a certain time period.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

mixing machines Apart from the number of vessels, also the number of mixing machines represents an important restriction. Especially when all resources of this kind are occupied and additionally a lot of batches need to be reworked, i.e. mixed again, another bottleneck emerges. This can result in long waiting times at the mixing machine which has a negative effect on the possible throughput. Every mixing machine can only be used for one vessel once at a time.

quality control All batches that are ready to be tested can be sent to the quality control right away. Thus, this is not a restriction per se, since there are no waiting times for the samples, but it still takes a certain time interval until it is completed. The duration of this testing stage depends on the product type. While the quality control is in progress, the batches stay in the vessels and are put on hold.

Although the quality control does not entail quantitative restrictions, the quality control is a crucial element since it approves a batch only with a certain probability and is thus an uncertainty within the process. The consequences this entails will be discussed in detail in chapter 2.4.

bottling station There is only a given number of bottling stations available which can be used by no more than one batch at a time. Setup times have to be adhered, especially when different product types succeed each other.

processing times The processing times for all tasks depends on the product type of the job only. In this discussion the availability and efficiency of manpower to operate the machines is omitted from consideration.

2.2.2 Objective of scheduling

In the problem at hand, the targeted value of the optimization is not obvious straightaway. For the considerations in this thesis only a specific part of a long process chain will be of interest. This part consists in producing intermediate products, i.e. different kinds of blending for further processing, and can be regarded as a preliminary stage to the actual production of silicone. In fact, no chemical reaction is part of this preceding stage, it is solely the preparation of the ingredients for the following stages which indeed involve chemical reactions. Thus, there seems to be no point in considering order properties such as due dates for optimization. It might seem reasonable to minimize monetary or time costs, though. Nevertheless, since this process is only a small preliminary stage for the rest of the production, the most intuitive is to aim at maximizing the throughput, or, as we will

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

look at it, minimizing the make span of a predefined set of orders. Due to the presence of uncertainty in the process, it will be of our concern to produce a feasible or better robust production plan. Thus, also in this problem the quantitative goal of aiming at processing a large number of jobs as fast as possible conflicts with qualitative goals such as obtaining a robust schedule.

2.2.3 Characteristics of the process industry

Since we have described the process in detail, we can now discuss whether and how or why not the characteristics presented in Jovan (2002) for problems in the process industry apply to the presented problem.

complex resource allocation Since a comparatively small number of input materials produces a large number of outputs, the mixing is of great importance. This leads to strictly following a high number of different allocation rules according to the specific recipes.

no intermediate storage option Once the vessel is filled, the blending has to stay in it until completion, i.e. until it is bottled in one of the tanks. There is no possibility to store it outside a vessel in the meantime. Nevertheless, the batch can be sort of stored in the vessel while it has to wait for the next operation to begin.

limited duration time The problem of a short shelf-live is especially common in the food and beverage industry. In this problem, there are no limitations given on how long the materials and blendings can be in the vessel. Nevertheless, there are two exceptions. Firstly, there might exist input materials, which cannot really be stored at all, which requires them to be processed right away or at least as soon as possible. Secondly, if a blending stays in a vessel for too long without being processed, it might be required that it is stirred once again before it is allowed to be bottled.

standards and regulations on traceability There are very strict regulations on the mixture of different materials at various points in the process. Every machine has to be cleaned or even sterilized before it can be used again to avoid residuals of the previous blending.

product flexibility If a blending is followed by a different product type, various changing times are unavoidable, for instance due to hygiene reasons as mentioned in the paragraph above. Therefore, the system can be described as not quite flexible when it comes to mingle different products or product types on one machine.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

uncertainty in production rate Especially due to the element of the compulsory quality check before bottling, some processing times are subject to uncertainty. This is one of the most difficult issues to tackle in the problem at hand. It is therefore our main concern in the rest of the thesis and we omit other typical uncertainties, e.g. of the demand.

2.3 Problem classification

With the definitions given in the introduction and the fact that we are blending different products together it can be said that the problem, which is discussed in this thesis, belongs to the process industry. Although there might is no chemical reaction involved in this part of the production process, this element is still found later on.

Since we are mixing together different kinds of input materials to obtain numerous kinds of products in the end, it is a so-called batch manufacturing production, according to the definition provided in Jovan (2002). In this kind of production it is typical to have several products share the same resources, i.e. machines and vessels (Jovan (2002)). In our case this applies to all products or at least all products for which the vessel, tank, etc. fulfill the requirements. After a task is completed on a resource, it is mandatory that it is cleaned before the next batch can use it. The setup time is sequence dependent, which means that it depends on the job just completed as well as on the job succeeding on the machine. It is zero or at least smaller if two batches of the same product type follow each other. As a result, the order in which the product types are scheduled will play an important role.

In order to classify our problem at hand either as a job-shop or as a flow-shop scheduling problem, we have to take a deep look into the structure of the production process. A shallow view on the production shows that every job has to take the same route through the process, starting with filling into vessels and being mixed until it is bottled into a tank, identifying it as a flow-job problem. Nevertheless, this is only true, if we consider the whole part with {mixing, quality check, rework} as one step of the process. Therefore, if we explicitly consider the recirculation through which some batches might have to go because of the quality control, then strictly speaking every job has an individual route through the process. This classifies the problem as a job-shop scheduling problem.

Regarding the usage of resources, although their suitability for the processing of a batch might be product type dependent, it can be assumed that for each task of a job there are several resources available, i.e. for every product type i there exists more than one vessel and more than one mixing machine that could be used for this type, if it is not occupied by another batch. This means that we have a production environment in which there are

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

machines in parallel, such that each job can be processed on exactly one of the machines, which belong to the same task and are suitable for the respective product type. It is important to consider, that in this case the schedule also has to provide an allocation of jobs to machines for each operation.

In summary, we can classify the problem as a so-called flexible job-shop production in the process industry, which is according to Maravelias (2012) defined as a job-shop problem combined with a parallel machines production environment.

2.4 The quality control as stochastic element

Since for the process of this discussion external dependencies are not given or have no effect on this part of the total production chain, the only source of uncertainty is given by the quality control, which is outsourced and therefore strictly speaking an external factor. Nevertheless, it affects the processing time of a job and consequently its further way through the process, which classifies it as a process inherent uncertainty.

Before looking at a formal description and possible solution procedures to the problem, once more the element of the mandatory quality check is examined in detail to emphasize the critical aspects as well as to define the main goals of this thesis.

First of all, one has to be aware of the consequences the quality control not only has on the respective job, that is approved or rejected, but on the whole schedule. According to the constraint presented in chapter 2.2.1, a job has to stay in the vessel for the whole process and if a batch gets rejected, rework needs to be done. This implies that this batch needs to be assigned to a mixing machine again and occupies the vessel for at least another mixing time + testing time periods. Additionally, if the time it takes until the batch finally reaches the bottling station is not as originally expected, it cannot be bottled at the point in time for which it was originally scheduled. This will probably result in a period, in which a bottling station is not used at all and instead in a period where more batches are waiting to be bottled than machines are available.

Thus, the question that we want to discuss in this thesis is how to create (and maybe adapt dynamically) a schedule, which considers the uncertainty in such a way that a schedule is not immediately infeasible once a batch has to be tested again, or at least more often than expected, but is still efficient and as optimal as possible. The optimality is represented by the total make span of the process for a predefined set of jobs.

In the following chapters, we will discuss the various possibilities to include uncertainty into a mathematical model. The analysis of these models should help us in finding the best way of dealing with the uncertain component in this real life scheduling problem.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

3 Model formulation of the problem

The first step of finding a solution to the scheduling problem is the construction of a (mathematical) model, which is an abstract and usually simplified representation of the real problem. There are many ways to formulate a problem into a model and those models can differ in quite a lot of features. In chapter 2 of Méndez et al. (2006) an overview of such features is provided. It is also emphasized that the chosen model representation and its inherent properties highly influence the time it takes to solve the model and therefore up to which problem size it is computationally reasonable to use it.

A major characteristic of a model for scheduling is the time representation. Depending on whether operations are allowed to start at any point within the time horizon or only at (the beginning of) certain time intervals, time can either be modeled continuously or discrete (Méndez et al. (2006)). In the discrete case the way of dividing the time horizon into periods and the granularity of these periods might significantly vary. Both approaches have advantages and disadvantages which will reflect especially in the computational performance of the model (Méndez et al. (2006)).

3.1 General solution requirements

Before turning to the detailed description of the proposed mathematical model, the conditions that need to be met in order to obtain a feasible schedule are discussed. We will emphasize on the most important properties that this job-shop problem includes apart from the typical time and disjunctive constraints found in these kinds of problems as for example described in (Šeda, 2007, Chapter IV) or (Méndez et al., 2006, Chapter 3). In addition, we will discuss the assumptions that will be made in order to keep the model restrictions of manageable size and complexity.

3.1.1 Feasibility

By solving the mathematical representation of the optimization problem, we seek to find an optimal schedule determining at what time and on which machine the tasks of each job are performed. In order for the schedule to be realizable it has to fulfill basic conditions.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

- Every job goes through the whole process. This means that every task of a job is completed and it is done in the right order according to the operation plan.
- Every machine can only process one job at a time.
- Each task can only be processed by exactly one machine.
- The machine for each task has to be able to operate the respective operation.

3.1.2 Special solution requirements and properties

In a flexible job-shop problem typically multi-purpose machines are considered, which means that each machine is able to perform various operations (Jovan (2002)). In the problem at hand there are several machines available for each task but every machine has its predefined purpose, i.e. a mixing machine exclusively performs the task "mixing". As in all flexible scheduling problems, due to having the choice of operating machine out of a given set of resources, not only the sequence of the jobs' taks in the production process but also the assignment of these operations to a machine needs to be included into the model.

According to the problem description, setup times have to be considered. In many flexible job-shop problems found in the literature, these requirements are omitted in the assumptions made, such as in Fattahi et al. (2007). Usually, setup times are necessary because each machine is capable of various different operations and thus, especially if two succeeding jobs are scheduled for different operations on the same machine, it needs to be adjusted in between. In the considered problem, on the other hand, the necessity for setups originates in the cleaning of the machines after completing an operation to ensure hygiene standards. Since the time the cleaning takes depends on the product type which has been operated as well as on the product type that is operated next, the setup times depend on the sequence in which the tasks are scheduled on the particular machine. In this problem, the setup times are so-called separable setup times, which means that the setup can start as soon as the last task has finished, but does not have to wait until the next job is ready for the task (Maravelias (2012)).

Due to the requirement, that each batch has to stay in a vessel as long as it is being processed and the limited number of these vessels, the jobs being processed at the same time have to be restricted. Additionally, the model has to provide an assignment of jobs to the vessels.

The jobs are allowed to wait in the vessels before the start of the next task. This smooths the time constraints compared to modeling a strict no-wait policy.

A crucial property is the element of recirculation, which occurs due to the quality control that sends the batches back into the mixing stage, if they are not approved. In the

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

deterministic case, in which we assume a given task order for each job, these recirculations do not require to be included into the model explicitly, but are given implicitly by the operation plan. Later on, in the stochastic model formulation, the uncertainty arising from the number of recirculations is indeed considered.

3.1.3 Assumptions

The mathematical model is a representation of a real world process or situation. Therefore, one has the urge to ensure that the model is as close to realistic conditions as possible. Nevertheless, in order to keep the size and complexity within a certain scope, it is inevitable to make assumptions that simplify and reduce the model constraints. The assumptions made about this problem are listed in the following.

- All resources, i.e. vessels and machines, are available at all times. This means that we do not take maintenance or sudden breakdowns into consideration, which imply that a machine could not be used in a certain time interval.
- All machines used for the same operation perform equally, i.e. the process times only depend on the attributes of the jobs, but not on the assigned machine.
- There is no transfer time between operations. It implies that the next operation can start immediately after the previous one has finished and the setup is complete. This assumption is legit since all batches, once filled into a vessel, stay in there for the whole process and only the machines are directed to the assigned vessels to perform the operation.
- All jobs are solely defined by the product types which are produced. Thus, all job-related properties such as process times, setup times, etc. are equal for all jobs producing the same end-product.

3.2 Mixed Integer Linear Programming formulation

This chapter develops a deterministic Mixed Integer Linear Programming (MILP) formulation to our problem at hand in order to find an optimal schedule by providing the necessary information about the jobs and machines. This kind of problem includes continuous as well as integer decision variables and its objective function and the constraints are represented by linear functions (Grossmann et al. (1996)).

According to the problem description, the only stochastic element in the process is assumed to be the outcome of the quality control, which affects the number of required

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

recirculations. In the deterministic case, we assume that the number of recirculations for each job and thus the respective operation plans are taken as given.

The following optimization program is based on the model given by Mousakhani (2013), which ensures that the general feasibility conditions are met as stated above in (3.1.1). The model presented in this paper was designed to describe a general flexible job-shop problem, in which setup times are included. In models disregarding setup times, the binary variables define the allocation of tasks to machines for each task of every job. As opposed to this, the explicit consideration of setup times in the model of Mousakhani (2013) is reflected in the fact that these binary decision variables are now defined with regards to consecutive tasks. This means that the allocations of the tasks are given through their sequence on one machine.

In Mousakhani (2013), the objective is to minimize the total tardiness, whereas in this problem the make span is minimized, which is defined by the maximum of the task end times. Thus, the variables measuring the tardiness of a job are omitted and the objective function is replaced by the make span.

In order to represent the problem at hand in reality, the model has been expanded by additional variables and constraints to represent the limited utilization of vessels. This has been done based on the corresponding parts of the model proposed in Caumond et al. (2009), in which they restrict the number of jobs simultaneously in the system. This corresponds exactly to the constraint, that at all times not more vessels can be occupied than there are available.

Since two completely unrelated models are combined, the notation of some parameters and variables have been adapted to enhance clarity and avoid ambiguity.

3.2.1 Parameters

In the problem at hand there are only four possible tasks, which are represented by numbers. Hence, the set of these tasks is given by

$$O = \{1\ (=\ \text{filling into vessels}),\ 2\ (=\ \text{mixing}),\ 3\ (=\ \text{testing}),\ 4\ (=\ \text{bottling})\}\,.$$

The set of jobs which are supposed to be scheduled is given by $J = \{1, 2, \ldots n\}$ with an operation sequence $O_{j,l} \in O$ for which $j \in J$ and $l = 1, 2, \ldots, l_j$ is the position of the operation in the sequence where l_j denotes the number of tasks the job j requires. For example, $O_{1,3} = 3$ states that the third task of job 1 is operation number 3, i.e. testing. Just as it is done in Mousakhani (2013), we also include a dummy job 0, whose task $O_{0,1} = 0$ needs to be processed once on each machine with a process time of zero, $p_{0,1} = 0$. This dummy job is needed to ensure a feasible assignment of jobs to machines which will

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

be discussed later on when the constraints are presented. The set of jobs including the dummy job is denoted by $J_0 = \{0, 1, 2, \ldots, n\}$.

The m machines available for production are given by the set $M = \{1, 2, \ldots, m\}$. In our case, each machine can only be used to perform exactly one type of operation $k \in O$. Thus, the machine set M is the union of the disjoint sets $M_k,\ k \in O$, of machines that are available for operation k : $\dot{\bigcup}_{k \in O} M_k = M$. On the other hand, for each job the tasks can only be performed on a certain subset of $M_k,\ k \in O$. We therefore define an eligibility parameter $\alpha_{j,l,i}$ for operation number l of job j, to determine whether or not the respective operation of this job can be performed on machine i.

$$\alpha_{j,l,i} = \begin{cases} 1, & \text{if } O_{j,l} \text{ can be performed on machine } i \\ 0, & \text{otherwise} \end{cases}, \qquad j \in J,\ l = 1, \ldots, l_j,\ i \in M$$

In general, the indices j, h will denote jobs, l, z operation numbers of the respective jobs and l_j, z_h the total number of operations of jobs j, h. The index i will be reserved for machines.

Each job has to be assigned to a vessel, in which all tasks are performed. The set of vessels is given by $V = \{1, \ldots, v^{\max}\}$, where $v^{\max}$ denotes the total number of vessels available.

In contrast to the model presented in Mousakhani (2013), since we assume that each machine has an equal operational performance, the processing times $p_{j,l},\ j \in J,\ l = 1, \ldots, l_j$ only depend on the operation $O_{j,l}$ and the job j, but not on the assigned machine.

We also want to include setup times. According to the assumptions made above, these setup times depend on the jobs succeeding on the same machine, which is called sequence dependent. Machines performing the same operations also have the same sequence dependent setup times, but these times might still vary for machines with a different purpose, e.g. for the same consecutive jobs, a mixing machine has a different setup time than a bottling machine. The setup times for job h immediately following job j on machine i is given by $set_{j,h,i}$.

Additionally, there is need to introduce the parameter L, which represents a large positive number. Including such a helper parameter is a typical way of dealing with constraints containing continuous and binary variables. As will be seen in the model, this parameter permits that certain constraints become relevant only in specific cases, whereas otherwise the property of L being a really large number ensures the fulfillment of the respective inequality.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

3.2.2 Variables

There are two kinds of variables, in particular continuous and binary variables. The continuous decision variables capture time values, whereas the binary variables describe assignments and sequences.

$$x_{j,l,h,z,i} = \begin{cases} 1, & \text{if } O_{j,l} \text{ is processed immediately after } O_{h,z} \text{ on machine } i \in M \\ 0, & \text{otherwise} \end{cases},$$

where $j \in J,\ l = 1, \dots, l_j,\ h \in J_0,\ z = 1, \dots, z_h$

$$a_{j,h} = \begin{cases} 0, & \text{if } h \in J \text{ is filled into a vessel before } j \in J \text{ is filled into a vessel} \\ 1, & \text{otherwise} \end{cases}$$

$$b_{j,h} = \begin{cases} 0, & \text{if } h \in J \text{ is bottled into a tank after } j \in J \text{ is filled into a vessel} \\ 1, & \text{otherwise} \end{cases}$$

$$w_{j,h} = \begin{cases} 0, & \text{if } h \in J \text{ is in a vessel when } j \in J \text{ is filled into a vessel} \\ 1, & \text{otherwise} \end{cases}$$

$$d_{j,v} = \begin{cases} 1, & \text{if job } j \in J \text{ is processed in vessel } v \in V \\ 0, & \text{otherwise} \end{cases}$$

$C_{j,l} \dots$ completion time for the l-th operation, $l = 1, \dots, l_j$, of job $j \in J$

$C^{\max} \dots$ make span, defined by the latest completion time among all n jobs

The variables x and d are the decision variables for the allocation of tasks to machines and jobs to vessels, respectively. The variable C results from the allocation of the operations and their processing times, while a, b and w are keeping track of the number of jobs simultaneously being processed and ensuring that the vessel restriction is satisfied.

3.2.3 Determinstic model

Objective

The objective is to minimize the make span $C^{\max}$. In order to assign a value to the variable $C^{\max}$, we include the following constraint.

$$C^{\max} \geq C_{j,l_j}, \qquad \forall j \in J \tag{3.1}$$

On the right-hand side is the completion time of the last task for each job, which is the completion time of the whole job. The make span is the smallest number such that the inequality holds for all jobs.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

Allocation constraints

The first set of constraints serves to ensure that the mapping of jobs and tasks to machines results in a feasible schedule. These constraints correspond to constraints (1)-(5) in Mousakhani (2013).

$$\sum_{h=0}^{n}\sum_{z=1}^{z_h}\sum_{i=1}^{m} x_{j,l,h,z,i} = 1, \qquad \forall j \in J,\ l = 1,\dots,l_j \tag{3.2}$$

This constraint states that every operation $O_{j,l}$ has exactly one predecessor on only one machine. More importantly, this ensures that every task is indeed planned because for all tasks l of each job j at least one binary variable $x_{j,l,.,.,.}$ has to be non-zero. This means that there is a fixed predecessor in the schedule and the assigned machine is given by the machine of its predecessor. On the other hand, the task cannot be planned more than once or on more than one machine. The dummy job ($h = 0,\ z_0 = 1$) has to be included because otherwise the first tasks of actual jobs assigned to a machine would not have a predecessor and would therefore not be scheduled properly.

$$\sum_{h=0}^{n}\sum_{z=1}^{z_h} x_{j,l,h,z,i} \le \alpha_{j,l,i}, \qquad \forall j \in J,\ l = 1,\dots,l_j,\ i \in M \tag{3.3}$$

The left-hand side can only be 1, if the right-hand side takes the value 1. Therefore, it can only exist a predecessor of $O_{j,l}$ on machine i if $O_{j,l}$ can actually be processed on this machine, which is represented by the value of $\alpha_{j,l,i}$ being 1. Otherwise, if $O_{j,l}$ cannot be processed on machine i, all variables $x_{j,l,.,.,.,i}$ are forced to be zero and $O_{j,l}$ is never assigned to machine i. Thus, the constraint ensures that to each operation an appropriate machine is assigned. The dummy job is included again, to ensure that this constraint also holds for the first actual operations on each machine.

$$\sum_{j=1}^{n}\sum_{l=1}^{l_j}\sum_{i=1}^{m} x_{j,l,h,z,i} \le 1, \qquad \forall h \in J,\ z = 1,\dots,z_h \tag{3.4}$$

Every task can have at most one successor. In this case only an inequality holds, because the last operation on machine i does not have a successor. With this constraint it is ruled out that two tasks are assigned to the same machine at the same time. Since the dummy job should be processed as the first operation on each machine and thus has a successor on

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

every machine, it is excluded from this constraint.

$$\sum_{j=1}^{n}\sum_{l=1}^{l_j} x_{j,l,0,1,i} \leq 1, \qquad \forall i \in M \tag{3.5}$$

This is a constraint to ensure that the dummy job is operated as the first operation on each machine.

$$\sum_{j=1}^{n}\sum_{l=1}^{l_j} x_{j,l,h,z,i} \leq \sum_{j=0}^{n}\sum_{l=1}^{l_j} x_{h,z,j,l,i}, \qquad \forall h \in J,\ z = 1,\dots,z_h,\ i \in M \tag{3.6}$$

The right-hand side takes only the values 0 or 1, otherwise task h of job z on machine i would have more than one predecessor. It only takes the value 1, if $O_{h,z}$ is assigned to machine i. Thus, there can only be a successor of $O_{h,z}$ on machine i if the task is actually planned on this machine. This ensures that two tasks can only be consecutive if they are planned on the same machine.

Sequence constraints

The next constraints consider the order in which tasks of the same job as well as two jobs assigned to the same machine are completed. These constraints are based on constraints (6)-(7) in Mousakhani (2013).

$$C_{j,1} \geq p_{j,1}, \qquad \forall j \in J \tag{3.7}$$

$$C_{j,l} \geq C_{j,l-1} + p_{j,l}, \qquad \forall j \in J,\ l = 2,\dots,l_j \tag{3.8}$$

This constraint states that the operation $O_{j,l}$ cannot end before its preceding operation $O_{j,l-1}$ has ended and the operation has been processed. For one, this secures the adherence of the processing times and also ensures that the tasks of job j are performed in the order specified.

It is important to note, that as opposed to the constraint given in Mousakhani (2013), the setup times are not included in this constraint. If the setup times were included into this constraint it would take the following form.

$$C_{j,l} \geq C_{j,l-1} + \underbrace{\sum_{h=0}^{n}\sum_{z=1}^{z_h}\sum_{i=1}^{m}(x_{j,l,h,z,i} \cdot set_{j,h,i})}_{(\star)} + p_{j,l}, \qquad \forall j \in J,\ l = 2,\dots,l_j$$

In this case, the constraint states that the setup cannot start before the preceding operation

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

is completed. Since we assume (see chapter 3.1.3) separable setup times we omitted the term ($\star$) is left out in this constraint. Nevertheless, the setup times are considered in the next sequence constraint.

$$C_{j,l} \geq C_{h,z} + \sum_{i=1}^{m} (x_{j,l,h,z,i} \cdot set_{j,h,i}) + p_{j,l} - L \cdot \left(1 - \sum_{i=1}^{m} x_{j,l,h,z,i}\right), \tag{3.9}$$

$$\forall j \in J,\ l = 1, \ldots, l_j,\ h \in J,\ z = 1, \ldots z_h$$

This constraint deals with the succession of two operations on a machine. If $O_{j,l}$ and $O_{h,z}$ are not scheduled on the same machine, the last sum on the right-hand side is zero, which means that because of the large value of L the constraint is fulfilled regardless of the values of the other variables. In this case, the constraint becomes irrelevant. If, on the other hand, the two operations are indeed assigned to machine i such that $O_{j,l}$ is scheduled after $O_{h,z}$, the sum becomes 1 and the large number is multiplied by 0, which means that it has no effect on the inequality any more. Now, the constraint ensures that the subsequent operation $O_{j,l}$ does not end before the previous operation $O_{h,z}$ has ended and the corresponding setup as well as processing times have passed.

Vessel constraints

So far, the constraints above describe a typical flexible job-shop scheduling problem with separable sequence dependent setup times. In the problem at hand there is one vital additional restriction that needs to be taken into account. Since there is only a limited number of vessels and each job stays in a vessel for the whole time it is being mixed or tested, we say that a job "enters the system" when it is filled into a vessel for processing, which is operation 1, and " leaves the system", when it is bottled into a tank, which is completed after operation 4. Now, we are able to rephrase the vessel constraint in such a way, that at all times the number of jobs in the system cannot exceed the total number of vessels available. In order to formalize this, we first introduce the value assignment of the auxiliary variables $a_{j,h}, b_{j,h}$ and $w_{j,h}$.

$$\forall j, h \in J,\ j \neq h:$$

$$(C_{h,1} - p_{h,1}) - (C_{j,1} - p_{j,1}) + L \cdot (1 - a_{j,h}) \geq 0 \tag{3.10}$$

$$(C_{j,1} - p_{j,1}) - (C_{h,1} - p_{h,1}) + L \cdot a_{j,h} \geq 0 \tag{3.11}$$

$$a_{j,h} + a_{h,j} \leq 1 \tag{3.12}$$

$$(C_{j,1} - p_{j,1}) - C_{h,z_h} + L \cdot (1 - b_{j,h}) \geq 0 \tag{3.13}$$

$$C_{h,z_h} - (C_{j,1} - p_{j,1}) + L \cdot b_{j,h} \geq 0 \tag{3.14}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

These constraints set the values for the variables $a_{j,h}$ and $b_{j,h}$. If the time at which h is filled into a vessel $(C_{h,1} - p_{h,1})$ is smaller than the time at which j is filled into a vessel $(C_{j,1} - p_{j,1})$, then $a_{j,h}$ has to be zero in order to avoid that the left-hand side of (3.10) is negative. On the other hand, if this does not hold, the left-hand side is positive no matter what value $a_{j,h}$ takes. In this case, inequality (3.11) ensures that $a_{j,h}$ has to take value 1, in order to fulfill the inequality. Thus, $a_{j,h}$ attains value 0 if and only if $(C_{j,1} - p_{j,1})$ is greater than $(C_{h,1} - p_{h,1})$, which means exactly, that j enters the system after h.

In an analogous way, constraint (3.13) ensures $b_{j,h} = 0$ if the time at which h is bottled into tanks C_{h,z_h} is greater than the time at which j is filled into a vessel $(C_{j,1} - p_{j,1})$. Otherwise, this constraint becomes irrelevant and constraint (3.14) sets $b_{j,h}$ to 1.

There is one more thing that needs to be considered in order to correctly assign values to a and as a result to w. In Caumond et al. (2009) there is assumed to be only one machine that brings jobs into or out of the system. In our case, we allow the number of filling as well as bottling machines to be greater than one. Therefore, it might occur that two different jobs, $j, h, j \neq h$, start the filling task at the exact same time. In this case, $\pm((C_{h,1} - p_{h,1}) - (C_{j,1} - p_{j,1})) = 0$ such that both constraints (3.10) and (3.11) become irrelevant because they hold for either value of $a_{j,h}$. Without any further restriction, the optimization sets the values $a_{j,h}$ and $a_{h,j}$ both to 1 because this corresponds to neither of the two entering the system after the other. This means that it is not considered in the rest of the constraints, that actually 2 vessels are occupied. To prevent this from happening we added the constraint (3.12), which ensures that it is always clear that one of the jobs is filled into a vessel before the other one.

Having defined the variables a and b, we can now use them to make sure that w attains the right value for every combination of jobs. This is done in the same fashion as constraints (15)-(17) in Caumond et al. (2009), in which the entering of the system is defined by the loaded trips going from the load or to the unload station.

$\forall j, h \in J,\ j \neq h:$

$$w_{j,h} - a_{j,h} - b_{j,h} \leq 0 \tag{3.15}$$

$$a_{j,h} - w_{j,h} \leq 0 \tag{3.16}$$

$$b_{j,h} - w_{j,h} \leq 0 \tag{3.17}$$

If $a_{j,h}$ and $b_{j,h}$ are both zero, this means that j is filled into a vessel after h has entered the system but before h has left the system. In this case, constraint (3.15) ensures that $w_{j,h} = 0$. If $a_{j,h}$ or $b_{j,h}$ is 1, i.e. when j is about to enter the system either h has not entered the system yet or has already left it, the left-hand side takes a negative value independent

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

of the value of $w_{j,h}$. Then, either (3.16) or (3.17) forces $w_{j,h}$ to be 1, otherwise the left-hand side is positive and the constraint violated.

Having ensured that $w_{j,h}$ only takes value 0, if h is currently in the system when j enters it, we can use the corresponding constraint (18) taken from Caumond et al. (2009) and adapt it to our notation. This constraint counts the number of jobs not in the system at the time j enters it, which for all j has to be greater than or equal to the total number of jobs minus the number of vessels available.

$$\sum_{h \in J,\ j \neq h} w_{j,h} \geq n - v^{\max}, \qquad \forall j \in J \tag{3.18}$$

At this point, it is guaranteed that at all times, there are at most v^{max} jobs filled into vessels. We also need to provide an assignment of vessels to jobs, assuring that every job has exactly one vessel and two jobs, which are at some point together in the system, do not use the same vessel. This is implemented by the following constraints.

$$d_{j,v} + d_{h,v} \leq w_{j,h} + w_{h,j}, \qquad \forall v \in V,\ j,h \in J,\ j \neq h \tag{3.19}$$

$$\sum_{v=1}^{v^{\max}} d_{j,v} = 1, \qquad \forall j \in J \tag{3.20}$$

Binary constraints

$$x_{j,l,h,z,i} \in \{0,1\}, \qquad \forall\ j \in J,\ l = 1,\dots,l_j,\ h \in J_0,\ z = 1,\dots,z_h,\ i \in M \tag{3.21}$$

$$a_{j,h} \in \{0,1\}, \qquad \forall j,h \in J \tag{3.22}$$

$$b_{j,h} \in \{0,1\}, \qquad \forall j,h \in J \tag{3.23}$$

$$w_{j,h} \in \{0,1\}, \qquad \forall j,h \in J \tag{3.24}$$

$$d_{j,v} \in \{0,1\}, \qquad \forall j \in J,\ v = 1,\dots,v^{\max} \tag{3.25}$$

Non-negativity constraints

$$C_{j,l} \geq 0, \qquad \forall j \in J,\ l = 1,\dots,l_j \tag{3.26}$$

$$C^{\max} \geq 0 \tag{3.27}$$

3.2.4 Complexity

In Garey et al. (1976) it is proven that job-shop scheduling problems with the objective of the shortest make span belong to the class of "NP-complete" problems. This is done

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

by proving this claim for the flow-shop scheduling problem and regarding the job-shop problem as a generalization, such that the results can be carried over. The fact that we are dealing with a flexible job-shop problem increases the complexity even more because the assignment of tasks to machines needs to be additionally considered. Problems of the "NP-complete" class have the property that it is possible to verify or refute an alleged solution in a reasonable amount of time, whereas there exists no efficient algorithm that can detect an optimal solution (Leung (2004)).

The following result holds regarding the number of variables in the model.

Lemma 1. In case that every job has the same number of tasks, $l_j^{max} = l^{max}$, $\forall j \in J$, there is a quadratic increase in the number of binary variables if either the number of jobs or the maximum number of tasks increase.

Proof. The allocation variable x is defined for all $j \in J$, $l = 1, \ldots, l_j$, $h \in J_0$, $z = 1, \ldots, z_h$, $i \in M$. Therefore, the model has

$$|M| \left(|J| \cdot l^{\max} \cdot (|J| \cdot l^{\max} + 1)\right) = |M| \left(|J|^2 \cdot (l^{\max})^2 + |J| \cdot l^{\max}\right)$$

allocation variables x, since the dummy job has only one task. If the number of jobs, $|J|$, or the number of tasks per job, $l^{\max}$, increases, it results in a quadratic increase in the number of variables. □

If we characterize the size of the problem by the number of its variables, **Lemma 1** shows that there is a quadratic increase in the problem size particularly whenever the number of jobs increases. It means that the problem size grows very fast if more jobs need to be scheduled and it will become computationally impossible to find a solution with exact methods once the number of jobs exceeds a certain (low) limit. As it will be discussed in chapter 3.2.6 this limit in our implementation of the deterministic model is already reached for no more than 10 jobs to be scheduled.

For many of the variables $x_{j,l,h,z,i}$ it is actually already known in advance, that the value has to be zero due to machine restrictions, for instance in the case of $O_{j,l} \neq O_{h,z}$. Thus, only a small subset of these variables is indeed relevant for the problem. Nevertheless, this huge number of variables is indispensable in order to develop a valid model that includes sequence dependent setup times but still has linear constraints.

3.2.5 Implementation in MATLAB

The deterministic model presented in the previous chapter has been implemented in MATLAB (MATLAB (2019))) with the package *Yalmip* (Löfberg (2004)) using the IBM CPLEX

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

12.10 solver (IBM (2019)). The complete source code of the function *detModel* performing the optimization according to the model above is provided in the appendix. This function has several input parameters:

n... number of jobs (including the dummy job)

scenario... vector containing the number of repetitions necessary for each job $j \in J$

v_max... number of vessels

m1, m2, m4... number of machines available for filling into vessels (m1), mixing (m2) and bottling (m4)

alpha... matrix containing the feasibility of machine $i \in M$ for each job $j \in J$ and operation $l = 1, \ldots, l_j$.

p... matrix of processing times for every task of each job $j \in J_0$; the processing time for the dummy job is zero

set... matrix containing the sequence dependent setup times for every machine and combination of succeeding jobs

Once the function is called, the value of L is set to a large number (10 000) and the total number of machines is calculated by m = m1 + m2 + v_max + m4. Since at all times the number of batches that can be tested simultaneously is unlimited, but we still need to assure that the task is scheduled, we include as many testing machines as vessels. This number is sufficient since only a job that is in a vessel can ever be ready for testing.

After that, the operation plans O for each job are created based on the scenario vector and the vector **l** is created, containing the number of tasks for each job. The parameter l_max is the maximum number of tasks a job has to go through.

Since all parameter values are now assigned, the optimization model is can be set up. First, all variables are initialized according to the *Yalmip* (Löfberg (2004)) syntax. It is not possible to set the index ranges for the operations depending on the jobs, so we have to create variables for the maximum number of each task for each job. In order to make sure that the result is still valid, we set the unnecessary variables to zero. The rest of the constraints then follows. At the end, the objective function is set to the variable C_max and the optimization problem defined before is solved using the IBM CPLEX 12.10 solver (IBM (2019)). The function then returns the make span C_max, the matrix of completion times C as well as the matrix of allocation variables x and d. By interpreting the information given by the function *detModel* the optimal schedule can be extracted. Additionally, not

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

only the allocation of tasks to machines is given by the solution but also the allocation of jobs to vessels, which is directly represented by the variable vector d.

Example 3.1

In order to show how to obtain a schedule from the information the function *detModel* delivers, we provide a small explanatory example. To keep parameter matrices and variables within manageable scales, we consider only 2 jobs that need to be scheduled. Nevertheless, the example still contains important elements that need to be considered in a feasible production schedule and is sufficiently complex to test the validity of the model and its implementation.

First of all, it is necessary to set the values for the input parameters the function *detModel* expects in order to solve the optimization problem. The number of jobs n is given by the number of actual jobs to be scheduled, which is 2, plus 1 to include the dummy job, i.e. $\mathsf{n} = 3$. We assume that job 1 is rejected by the quality control once, whereas job 2 is approved right away. Thus, the scenario vector is $[2, 1]$, because job 1 needs to go through operations 2 and 3 twice and job 2 only once. Next, the number of vessels is expected. We assume that $\mathsf{v_max} = 2$ such that both vessels can be processed at the same time. Furthermore, we choose the number of filling machines $\mathsf{m1} = 2$, the number of mixing machines $\mathsf{m2} = 1$ and the number of bottling machines $\mathsf{m4} = 1$. The number of testing machines is automatically be set to v_max, resulting in a total number of $\mathsf{m} = 6$ machines. In order to know which machines can be used by which jobs, we need to provide the eligibility matrix alpha. We assume that one filling machine is suitable for job 1, the other one for job 2. The testing machines can be used by both jobs and since for the other machines no alternative resources exist, both jobs can be processed on those. The 3-dimensional matrix alpha consists of the two 2-dimensional matrices, for each job.

$$\mathsf{alpha}\{1\} = \begin{array}{c} \begin{array}{cccccc} \text{mach 1} & \text{mach 2} & \text{mach 3} & \text{mach 4} & \text{mach 5} & \text{mach 6} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \begin{array}{l} \\ \text{op 1} \\ \text{op 2} \\ \text{op 3} \\ \text{op 4} \end{array},$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

$$\mathsf{alpha}\{2\} = \begin{array}{c} \begin{array}{cccccc} \text{mach 1} & \text{mach 2} & \text{mach 3} & \text{mach 4} & \text{mach 5} & \text{mach 6} \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \begin{array}{l} \\ \text{op 1} \\ \text{op 2} \\ \text{op 3} \\ \text{op 4} \end{array}$$

Note that the two matrices only differ in the first two columns of the first row, since we assumed that each of the jobs is only allowed to use one of the filling machines.

The next input parameter are the processing times, which are given in the table below. Since the processing times do not depend on the chosen machine, we only need to provide the times for every tasks of each job. The processing times for the dummy jobs are set to zero.

$$\mathsf{p} = \begin{array}{c} \begin{array}{cccc} \text{op 1} & \text{op 2} & \text{op 3} & \text{op 4} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 \\ 2 & 1 & 1 & 2.3 \\ 1 & 4.5 & 3 & 1.7 \end{pmatrix} \end{array} \begin{array}{l} \\ \text{job 0} \\ \text{job 1} \\ \text{job 2} \end{array}$$

The last information that needs to be provided are the setup times. This is again a 3-dimensional matrix, containing the setup times depending on the preceding job, the succeeding job and the machine. This is also given for the dummy job, such that each machine can have a preparation or cleaning time before the first actual job is processed. The setup times have to be set for each machine separately. To keep it as simple as possible, we assume that the following setup matrix is valid for every machine.

$$\mathsf{set} = \begin{array}{c} \begin{array}{ccc} \text{job 0} & \text{job 1} & \text{job 2} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 2 \\ 1 & 2 & 0.5 \end{pmatrix} \end{array} \begin{array}{l} \\ \text{job 0} \\ \text{job 1} \\ \text{job 2} \end{array}$$

This matrix states that if the same jobs succeed each other on one machine, their setup time is only 0.5 time units whereas if two different jobs follow each other, e.g. job 2 succeeds job 1 or the other way round, then the setup time is significantly higher with 2 time periods. The initial setup for each machine is given by the values in the first column. In this case the values for setups after job 0 are all given by 1 independent of the first job that is actually processed on the respective machine.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

The size of the input parameters increases with the number of jobs and machines. Especially the setup time and eligibility matrices become quite cumbersome for large problems.

Providing the information above to the *detModel* function, we obtain the values for the variables C, $C^{\max}$, x and d as matrices makespan, schedule, allocation and vesselallocation. With those values and matrices we are able to create a so-called Gantt-Diagramm to visualize the optimal production plan.

Gantt-diagramm The Gantt-diagramm is named for the economist H.L. Gantt and is a graphical representation of the assignment of all operations to machines and other resources as well as to the time interval in which they are processed (Iwainsky and Wilhelmi (1994)). Typically, each horizontal line corresponds to the chronologically ordered tasks processed on one machine over time, which might be operations belonging to different jobs (Iwainsky and Wilhelmi (1994)). In order to associate tasks of the same job, they have matching hatching patterns or the same color.

If we want to construct a Gantt-diagramm to visualize the production schedule in the case of **Example 3.1**, we need the values of the variable allocation which indicate the sequence of the jobs on each machine. The matrix of optimal completion times schedule with the respective processing times can be used to calculate the time intervals in which the operations are performed. In this simple case, the Gantt-diagramm can almost completely be constructed by looking at the completion times table only.

$$\text{schedule} = \begin{array}{c} \begin{array}{cccccc} \text{op 1} & \text{op 2-1} & \text{op 3-1} & \text{op 2-2} & \text{op 3-2} & \text{op 4} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & - & - & 0 \\ 3 & 9.5 & 10.5 & 11.5 & 12.5 & 15.5 \\ 2 & 6.5 & 9.5 & - & - & 11.2 \end{pmatrix} \end{array} \begin{array}{l} \\ \text{job 0} \\ \text{job 1} \\ \text{job 2} \end{array}$$

Especially in more complex problems, the information provided by the matrix allocation is needed in order to know exactly to which machines the operations are assigned, since the completion times might produce an ambiguous schedule. In this case, we would also need to take a look at the variable allocation to know which job uses which testing machine. Although this allocation is in fact irrelevant because the testing is done outside of the process, by looking at the matrices for machines 4 and 5 of the variable allocation we see for example, that the entry at allocation(2,3,1,1,4) takes the value 1, which means that the testing operation of job 1 is the direct successor of the dummy job 0 (for indexing reasons the job 0 corresponds to job 1 and job 1 to job 2 in MATLAB (MATLAB (2019))) on machine 4. Thus, we know that the operation is scheduled to be on this machine. The

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

matrix vesselallocation provides the additional information which vessel is designated to process which job. In this case, job 1 is assigned to vessel 1 and job 2 to vessel 2.

The Gantt-diagramm for the optimal schedule with a minimum makespan of 15.5 is shown in figure 3.1 below.

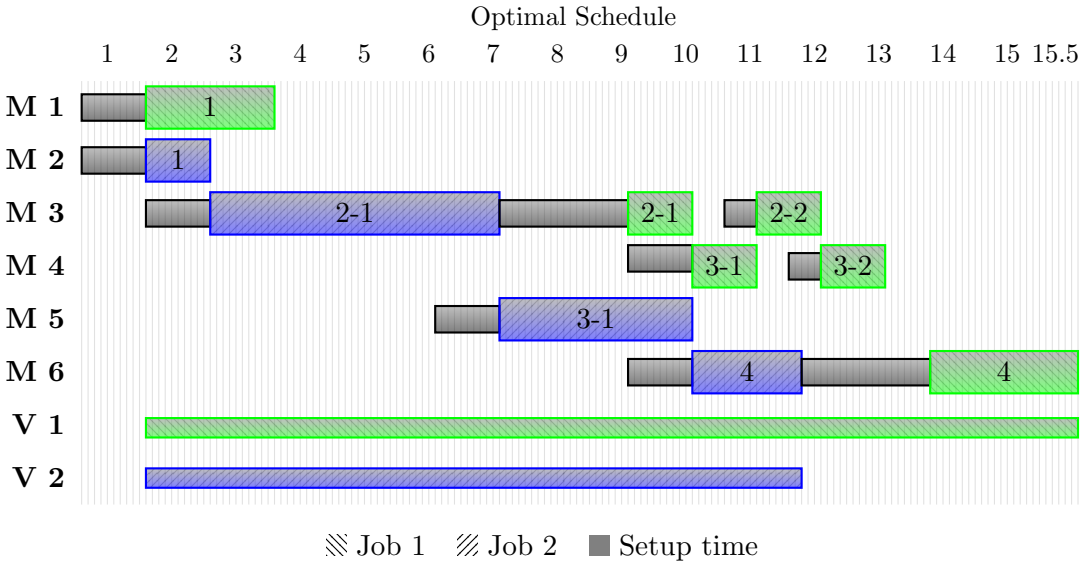

Figure 3.1: The optimal schedule for **Example 3.1**, own drawing

The Gantt-diagram shows, that the function *detModel* actually finds a feasible schedule. All tasks are scheduled in the right order on an eligible machine and every machine processes only one job at a time. All setup times are considered either with the length of 1 if it is the first operation on the respective machine, with length 2 if two different jobs succeed on one machine or of length 0.5 if operations of the same jobs succeed. In order so minimize the make span the solution makes use of the fact that the setup times can start as soon as the operation on the machine is done but do not have to wait until the previous task of the succeeding job on the machine is done. It is also important to note that on machine 3 the optimal solution does not split up the tasks of job 1 but instead schedules job 2 first and afterwards the tasks of job 1 consecutively. This is due to the different setup times, because if the task of job 2 was scheduled between the tasks of job 1, it would cause twice a setup times of 2 instead of one of 2 and one of 0.5.

In the Gantt-diagram we can also see in which time periods job 1 is processed in vessel 1 and job 2 in vessel 2. This information is represented by the bottom two lines.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

3.2.6 Computational limits

The IBM CPLEX 12.10 solver (IBM (2019)) is a powerful solver. Nevertheless, a MILP problem such as this one poses a big computational challenge due to the combinatorial complexity. If we set the number of vessels as well as the number of machines for each operation to 2 and only vary the number of jobs we can see the following performance of the implementation. We do not consider any rework procedure for any jobs. For small problem sizes up to 8 actual jobs, the solution to the deterministic model implemented in the function *detModel* is found within seconds up to minutes. Once the number of jobs has reached 9 without the dummy job, it already takes two hours. For more than 9 jobs to schedule the solver is no longer able to provide a solution within a reasonable amount of time.

It has to be noted that in general the time it takes to find a solution not only depends on the number of jobs but also on the number of vessels and machines relative to the number of jobs. For example, for 7 real jobs, a solution with a setting of 4 available vessels and two machines each is found within a few minutes, whereas for 5 vessels it already takes around 50 minutes.

3.2.7 Applicability to problems of real world dimensions

Since the number of jobs that can be scheduled in a reasonable amount of time is quite small, exact methods for solving the deterministic approach cannot be deployed to problems of real world size with hundreds and thousands of jobs. In order to still use this model as well as the models in the next chapters that build on this one, it will be necessary to resort to heuristic methods or hybrid methods. The latter is a combination of exact and heuristic approaches.

The CPLEX solver 12.10 (IBM (2019)) for the implementation of the model uses a Branch & Bound algorithm which lacks in performance due to the feasible solutions being only found deep down in the Branch & Bound tree (Kallrath (2002)). Heuristic algorithms, on the other hand, can exploit the particular problem structure and would therefore able to solve problems such as this deterministic model in an efficient amount of time. Examples for such solution approaches are simulated annealing and tabu search as applied for instance in Fattahi et al. (2007) to find a solution to a flexible job-shop problem. In Mousakhani (2013), on which the deterministic model is based, an iterated local search algorithm is proposed. The local search procedure, that tries to improve the current solution within its neighborhood, is combined with a migration algorithm, that explores the rest of the solution space. The migration feature is important to avoid the problem of being trapped in a local optimum. Thus, apart from their great performance, it must not be neglected

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

that heuristic approaches, in contrast to exact methods, cannot guarantee that an actual optimal solution is found nor how far the objective is from the actual optimal value (Méndez et al. (2006)).

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

4 Robust approach

This chapter presents the first approach that integrates the uncertainty into the model. Before this will be done, we start with an example that motivates taking into account the uncertainty in the process. Then, based on the deterministic model provided in the previous chapter, we will discuss robust model approaches. In order to do so, we first discuss how to represent the uncertainty in the model. After a short introduction of a metric for measuring the robustness of a given model, we will present two different approaches to robust scheduling. Both approaches will be described based on a generic model and we will attempt at applying them to our problem. Although it will turn out that the first approach does not deliver any actual new insight for the problem at hand, describing how to construct a robust version of a generic MILP will provide a basic understanding of the idea behind robust scheduling.

4.1 Motivation

Example 4.1

To show the limitations of the applicability of deterministic approaches to our problem, we will introduce the following example. This example is as simple and small as possible but already able to show how the deterministic solution deteriorates in case of unexpected but possible turn of events. We will assume that there are 3 jobs to be scheduled and for now, that all jobs pass the quality control the first time with a probability of 1. Hence, all jobs have to follow the operation sequence [1, 2, 3, 4]. The processing times for the jobs are listed in table 3.1 below. In order to keep this example simple, we will ignore any setup times. There are only 2 vessels available, which implies that not all jobs can be processed at the same time. Furthermore, there will be only one machine for each operation, except for testing, which has no limit on the number of jobs being processed simultaneously.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

	Filling	Mixing	Testing	Bottling
Job 1	1	2	2	1
Job 2	2	4	2	1
Job 3	2	2	2	2

Table 4.1: Processing times for **Example 4.1**

4.1.1 Deterministic solution

The resulting optimal schedule provided by the implementation of the deterministic model in MATLAB (MATLAB (2019)) is presented below.

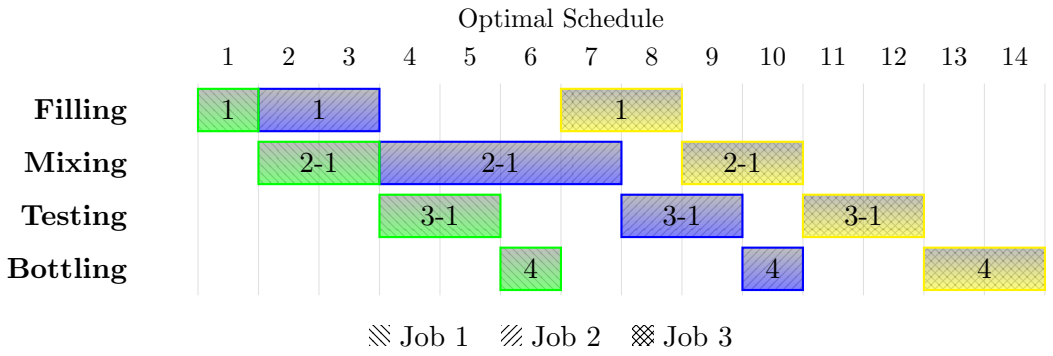

Figure 4.1: The optimal schedule for **Example 4.1**, own drawing

Figure 4.1 depicts the optimal schedule as found by the IBM CPLEX 12.10 solver (IBM (2019)) with the minimal make span of 14 time periods. It can be seen that every operation of each job is scheduled in the right sequence 1-4 and no two operations are scheduled on the same machine at the same time. Additionally, job 3 (crosshatch pattern) cannot enter the system until one of the other jobs has left it, because the number of vessels is restricted to 2. This is why the first operation of job 3 starts at the beginning of the 7-th time period, which is exactly the moment in which job 1 (north-west lines) leaves the system.

It can also be interpreted that job 2 (north-west lines), which takes a long time for mixing, is the second job to enter the system. This seems to be the optimal way of scheduling, because if job 2 was the first job to start operating, it would postpone the completion time of job 1. This is due to the fact that job 2 occupies the filling machine and therefore job 1 would not be able to enter the system before the third time period. Also, since the total sum of processing times for job 2 is 9, job 3 would never be able to start in the 7-th period. This will lead inevitably to a larger make span. If job 2 was scheduled as the last one to enter the system, it would not be able to enter the system before job 1 has left it, which

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

is at time period 7. Since the sum of processing times of job 2 is greater than the one of job 3, also this processing order would lead to a schedule with a sub-optimal objective value. Changing the order of job 1 and 3, on the other hand, would have an effect on the scheduling times of all three jobs but would lead to the same minimal make span of $C^{\max} = 14$.

4.1.2 Impact of quality control's outcome on production schedule

To show an important impact the outcome of the quality control has on the solution, we will start with the example above and assume that job 1 is rejected by the quality control the first time it is tested. This means that operation 4, which is scheduled to start in period 6, is no longer able to be performed at this point in time, because the moment the job is rejected, the operation sequence is extended by another round of operations 2 and 3. Instead of being bottled into a tank, at the beginning of period 7, job 1 needs to use one of the machines designated for "Mixing" again. In this example we only have one such machine available and according to the schedule obtained from the deterministic model job 2 is occupying the mixing machine until period 7. Thus, job 1 has to wait until it can be processed again. The only reasonable thing to do is scheduling the 4-th operation of job 1 to start in period 9, immediately after job 2 is done. This results in changing the whole schedule afterwards, which cannot be avoided anyway because there is no vessel available for job 3 before job 2 has completed its production process. Thus, operation 1 of job 3 needs to be postponed and therefore all subsequent operations. This will then result in a schedule as depicted in figure 3.2. The operations that cannot take place at the time they were originally scheduled are highlighted by a red border.

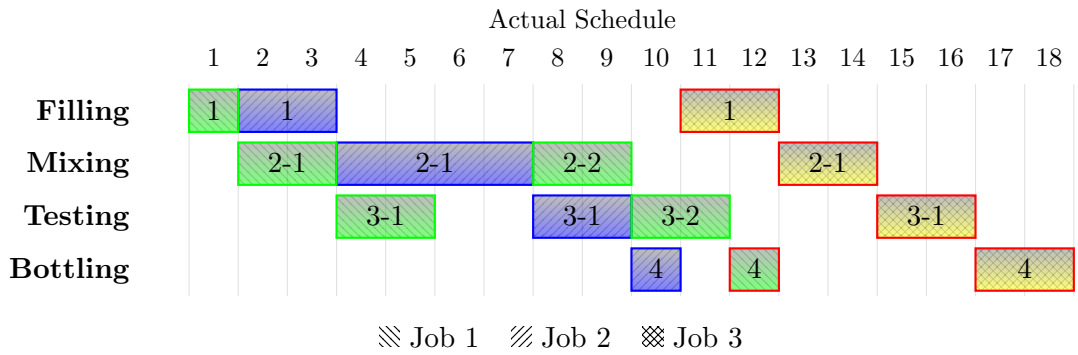

Figure 4.2: The realization of the schedule for **Example 4.1** in case job 1 gets rejected once, own drawing

Solving this problem with the deterministic model but already including the reworking

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

necessary for job 1 gives a schedule in which job 3 starts the production process, job 1 follows and job 2 is the last job to start the processing. This schedule has a make span of 17. This might not seem like a big difference in production time, but in problems of real world dimensions large additional production time might be the result of an unwisely chosen schedule.

This example shows, that if a stochastic element such as the outcome of the quality control attains a different value than originally expected, the schedule can become infeasible. For example, job 3 cannot be filled into a vessel in time period 7 according to the optimal schedule depicted in figure 3.1 because it would violate the vessel restriction constraint. Also, it will no longer hold, that job 3 is the immediate successor of job 2 on the mixing machine. This means that without further adjustments to the schedule, it cannot be executed. The scenario described in this example serves as a motivation to explicitly take into account the uncertainty inherent to the process. Thus, the main goal is to obtain a solution which needs no or as less adjustments as possible to actually execute the schedule.

4.2 Representing uncertainty in the model

Typically, to obtain a stochastic model, the uncertain parameters are replaced by stochastic variables representing the corresponding distributions. In the problem at hand, the stochastic element is the uncertain outcome of the quality control but it is at first not so clear how this can be reflected in the model. It is important to realize how the outcome of the quality control affects the parameters. In fact, it only influences the operation sequence of the jobs, because every job needs as many repetitions of operations 2 and 3 as the number of times the quality control rejects the respective batches. This is already a crucial complication compared to typical job shop problems found in the literature which take the operation sequences as fixed and given. Therefore, we need to find a way to represent the uncertainty occurring in the operation sequence through parameters of the model.

One way of doing so is to consider the processing times of operations 2 and 3 as random variables, such that they are either positive if the operation needs to be considered or zero, if the job has already passed the quality control and thus does not need to be reworked. This means, that we obtain discrete random variables taking only 2 possible values. Additionally, the time variables C as well as the make span $C^{\max}$ become random variables because of the implications the result of the quality control has on the schedule. To make this whole idea more clear, we provide a short example.

Let us assume that it is known from historical data that a job j does typically not need more than 2 tries to get approved by the quality control, which means that the probability for not being approved at least by the second time it is tested is insignificantly

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

small. We will therefore consider an operation sequence that includes two tries, i.e. two times the mixing and testing operation, which looks like this: $[1, 2, 3, 2, 3, 4]$. We will assume that the processing time of job j for the operation "mixing" is 3 periods and for "testing" 2 time periods, i.e. $p_{j,2} = 3,\ p_{j,3} = 2$. Since every job needs to go through those operations at least once, these processing times are not random variables but are fixed. As opposed to this, $p_{j,4}$ and $p_{j,5}$ are random variables, because if the first testing approves, there is no need to "activate" the 4-th and 5-th operation of the operation plan, which is represented by processing times equal to zero, $p_{j,4} = p_{j,5} = 0$. Thus, technically, the operations are scheduled in the plan, but are irrelevant since they do not consume any time. Nevertheless, one has to make sure that the corresponding setup times are zero as well. If the quality control does not approve the batch the first time, it will actually need a rework and the processing times are equal to the processing times of the first time, i.e. $p_{j,4} = p_{j,2} = 3,\ p_{j,5} = p_{j,3} = 2$. Thus, $p_{j,4},\ p_{j,5}$ are discrete random variables with possible values in $\{0, 3\}$ or $\{0, 2\}$, respectively.

Including the outcome of the quality control in such a way as just described reduces the problem to a scheduling problem with uncertain processing times, which is considered in the literature various times, for example in Balasubramanian and Grossmann (2002).

In general, an easy and intuitive way of dealing with random variables as model parameters is to use their mean values and thus to obtain a deterministic model again. There will be many cases in which this solution is far from optimal, as the example in Balasubramanian and Grossmann (2002) shows, because the found schedule might be sub-optimal with respect to uncertainty and underestimate the expected value of the make span. In the problem at hand, this does not seem like a promising strategy either, since each random variable has only two possible values.

4.3 Robust scheduling

The main goal of robust scheduling is to find a schedule that is build in such a way, that disruptions, i.e. parameters attaining different values, not only have as small effects as possible on the value of the objective but also try to accomplish that the predicted schedule does not differ too much from the actually realized one (Li and Ierapetritou (2008)). A major advantage of this approach is that no probability distribution for the stochastic variables needs to be found, but instead different potential values are considered (Li and Ierapetritou (2008)).

With regards to robust scheduling, two types of robustness can be distinguished. A schedule is called "solution robust", if in all scenarios the value of the objective function is close to the optimum, whereas "model robustness" describes the property of a schedule to

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

be feasible for various realizations of the uncertain parameters (Li and Ierapetritou (2008)).

4.4 Metrics for robustness

For a given deterministic model, it might be of interest to evaluate the performance of a resulting schedule when different scenarios realize in order to assess the robustness of the solution to perturbations. The most common measure is given by the standard deviation, defined as follows according to Li and Ierapetritou (2008). We consider the set of k scenarios, represented by different parameter values. Starting with the given schedule, which provides the order in which the jobs and tasks will start on the machines, the make span H_k that will turn out to realize in each scenario k is computed. The standard deviation is then given as

$$\sigma = \sqrt{\sum_k \frac{(H_k - H_{\text{avg}})^2}{p_{\text{tot}} - 1}}. \tag{4.1}$$

$p_{\text{tot}} \ldots$ total number of scenarios
$H_{\text{avg}} = \frac{\sum_k H_k}{p_{\text{tot}}} \ldots$ average make span

This measure of robustness can only be used if all scenarios are feasible in order to compute a make span. To be able to obtain an appropriate measure in case of infeasible scenarios as well, the corrected standard deviation can be used (Li and Ierapetritou (2008)). This metric distinguishes between feasible and infeasible realizations. In case of infeasibility, we will assume that first the schedule tries to process as many tasks as possible, then the schedule is started a second time to process the rest of the operations. Thus, the total make span H_k is given by the make span for the optimal but not complete production plus the make span for the orders not completed in the previous schedule. Then, the definition of the standard deviation (4.1) can be used as well.

4.5 Robust model 1

The following approach is based on the one described in Lin et al. (2004). In this thesis, the basis for the robust model is the generic MILP formulation

$$\min_{x,y} / \max \quad c^T x + d^T y \tag{4.2}$$

$$\text{s.t.} \quad Ex + Fy = e \tag{4.3}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

$$Ax + By \leq p \tag{4.4}$$

$$\underline{x} \leq x \leq \bar{x} \tag{4.5}$$

$$y \in \{0, 1\}, \tag{4.6}$$

where x, y are the decision variables, c, d, e, p parameter vectors and E, F, A, B parameter matrices representing the constraints. It is assumed that the elements of A and B as well as of p are subject to random perturbations, such that the uncertain coefficients $\tilde{a}, \tilde{b}$ and $\tilde{p}$ might attain any value within the following intervals.

$$|\tilde{a}_{lm} - a_{lm}| \leq \epsilon |a_{lm}|, \ |\tilde{b}_{lk} - b_{lk}| \leq \epsilon |b_{lk}|, \ |\tilde{p}_l - p_l| \leq \epsilon |p_l|, \qquad \forall l, m, k \tag{4.7}$$

robust In Lin et al. (2004) a solution (x, y) is called robust, if the following two conditions are fulfilled.

- The solution is feasible for the original model, (4.2)-(4.6).
- Let δ denote an infeasibility tolerance. The l-th inequality constraint has to hold such that the error does not exceed $\delta \cdot \max[1, |p_l|]$, regardless of the values the coefficients a, b as well as the parameter p attain within the intervals (4.7).

In order to find a robust version of the MILP model presented above, the original problem is expanded by a new set of constraints. Those constraints are the inequalities (4.4) assuming the worst case scenario for every uncertain parameter allowing for the tolerance δ. Thus, the solution to this formulation is feasible if the uncertain parameters attain their nominal values, since all original constraints are still included, and also fulfills the robustness conditions due to the new constraints. We will now describe the procedure for obtaining the robust version of the MILP model presented above by following the argumentation within the proof of ***Theorem 1*** in Lin et al. (2004).

First of all, we divide the set of indices for the variables x as well as y into the ones for which the coefficients are uncertain and the ones for which they are not. The sets of parameters subject to random deviations are given by M_l and K_l for the variables x and y, respectively. For each index in M_l or K_l we replace the coefficient by the random variable denoted as above by $\tilde{a}$ and $\tilde{b}$. Also, the parameter p on the right-hand side of the inequality is substituted by its stochastic counterpart $\tilde{p}$. Thus, the l-th constraint will be written as

$$\sum_{m \notin M_l} a_{lm} x_m + \sum_{m \in M_l} \tilde{a}_{lm} x_m + \sum_{k \notin K_l} b_{lk} y_k + \sum_{k \in K_l} \tilde{b}_{lk} y_k \ \leq \ \tilde{p}_l + \delta \max[1, |p_l|]. \tag{4.8}$$

We already added the term $\delta \cdot \max[1, |p_l|]$, which is the maximum value by which the constraint is allowed to be violated. Note that in this construction δ is equal in all inequalities.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Of course, it is also possible and might be reasonable to choose individual values for each inequality.

In order to find a robust solution according to the definition above, it has to fulfill the inequality (4.8) no matter which values the parameters $\tilde{a}, \tilde{b}$ and $\tilde{p}$ attain within the respective intervals (4.7) in the end.

Therefore, the next step is to identify the values of the parameters in the worst case. For the left-hand side of inequality (4.4) the worst case are the largest possible values.

$$\tilde{a}_{lm}x_m = a_{lm}x_m + \epsilon|a_{lm}||x_m|, \tag{4.9}$$

$$\tilde{b}_{lk}y_k = b_{lk}y_k + \epsilon|b_{lk}|y_k, \tag{4.10}$$

For the uncertain parameters on right-hand side of (4.4) the lowest possible value is the worst case.

$$\tilde{p}_l = p_l - \epsilon|p_l| \tag{4.11}$$

These worst case values for the coefficients are now substituted into constraint (4.8), yielding the following constraint.

$$\begin{aligned} & \sum_{m \notin M_l} a_{lm}x_m + \sum_{m \in M_l} (a_{lm}x_m + \epsilon|a_{lm}||x_m|) + \sum_{k \notin K_l} b_{lk}y_k + \sum_{k \in K_l} (b_{lk}y_k + \epsilon|b_{lk}|y_k) \\ & \leq p_l - \epsilon|p_l| + \delta \max[1, |p_l|] \\ \Leftrightarrow \quad & \sum_{m} a_{lm}x_m + \epsilon \sum_{m \in M_l} |a_{lm}||x_m| + \sum_{k} b_{lk}y_k + \epsilon \sum_{k \in K_l} |b_{lk}|y_k \\ & \leq p_l - \epsilon|p_l| + \delta \max[1, |p_l|] \end{aligned}$$

Adding this constraint for all l to the original model yields a robust formulation.

4.5.1 Application to the problem

We will now apply this procedure to the problem at hand using the model formulation from chapter 3.2.3 and representing the uncertainty via uncertain processing times as described in chapter 4.2. The first step is to identify the relevant constraints for the robustness considerations. Those constraints are the ones containing processing times that are of stochastic nature. The only constraints containing processing times are (3.7)-(3.14). Since every job has to enter the system by being filled into a vessel and leaves the system by being bottled into a tank exactly once, constraints (3.8) as well as (3.10)-(3.14) contain

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

exclusively deterministic processing times and thus do not have to be considered in a robust formulation. The only two constraints that are of relevance are the sequence constraints (3.8) and (3.9). We will now develop the inequalities that have to be added to the original problem in order to obtain a robust formulation.

The stochastic processing times $\tilde{p}_{j,l}$ of the rework operations are given by the index set $P := \{(j,l) : j \in J,\ l > 3,\ O_{j,l} \in \{2,3\}\}$. Each of these parameters, denoted by $\tilde{p}_{j,l}, (j,l) \in P$, will either attain the value 0 or the value $p_{j,2}$ or $p_{j,3}$ depending on the operation $O_{j,l}$, where the later case can be equivalently expressed as $\tilde{p}_{j,l} = p_{j,O_{j,l}}$.

We seek inequalities securing that for each realization 0 or $p_{j,O_{j,l}}$ of the stochastic variables $\tilde{p}_{j,l}, (j,l) \in P$ the constraint will still hold. In each inequality of (3.8) as well as (3.9) only one processing time appears. This is the processing time of the operation corresponding to the completion time on the left-hand side of the inequality. Therefore, we only need to add constraints for the inequalities for which the processing times are represented in the set P. For those, we assume the worst case $\tilde{p}_{j,l} = p_{j,O_{j,l}}$ and insert this into the constraints.

$\forall (j,l) \in P:$

$$C_{j,l} \geq C_{j,l-1} + \sum_{h=0}^{n} \sum_{z=1}^{z_h} \sum_{i=1}^{m} (x_{j,l,h,z,i} \cdot set_{j,h,i}) + p_{j,O_{j,l}} - \delta \max\{1, |p_{j,O_{j,l}}|\} \tag{4.12}$$

$\forall (j,l) \in P,\ \forall\ h \in J,\ z = 1, \dots z_h:$

$$C_{j,l} \geq C_{h,z} + \sum_{i=1}^{m} (x_{j,l,h,z,i} \cdot set_{j,h,i}) + p_{j,O_{j,l}} - L \cdot \left(1 - \sum_{i=1}^{m} x_{j,l,h,z,i}\right) - \delta \max\{1, |p_{j,O_{j,l}}|\} \tag{4.13}$$

The infeasibility tolerance δ has been added to the constraints as well. In this case, it needs to have a negative sign, since it is placed on the other side of the inequality. Expanding the deterministic model from chapter 3 by constraints (4.12) and (4.13) will result in a robust formulation of this model.

4.5.2 Remarks to the approach

The way the infeasibility tolerance δ is added to the robust constraints in Lin et al. (2004) has to be called into question. Changing the units of the variables in the original constraint, i.e. multiplying it with a constant, does not change the feasible region of the problem. However, in the robust variant of the constraint this might indeed happen in the case that $|p_l| < 1$ has held before but is due to the multiplication now $|p_l| > 1$. Then, the feasible

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

region is different than originally due to the fact that δ is multiplied by $\max\{1, |p_l|\}$. Thus, it might seem more reasonable to replace this factor by, for instance, $|p_l|$ alone.

When looking at the constraints added to the robust formulation of the deterministic model, it is important to notice that each constraint for fixed values of j, l, h, z does only contain one stochastic processing time. This means that for $\delta = 0$ this robust formulation is equivalent to the worst case formulation of the deterministic model, in which the processing times are set to the largest value they are able to attain. Thus, the resulting robust model - apart from the infeasibility tolerance - does not really offer a new way of finding a solution.

4.6 Transformation of deterministic model

The very same argument that only one processing time appears in each constraint is discussed in Shafia et al. (2011). They consider a MILP model of a flexible job-shop problem whose constraints subject to uncertainty resemble our constraints (3.8) and (3.9). As a result, the consequences of varying values for the processing times are only relevant among consecutive tasks either within a job or for the ones scheduled on the same machine. This is a major weakness of the approach presented above in the case of uncertain processing times, since it does not take into account the effects of perturbations on all succeeding tasks but only on the consecutive ones. In order to resolve this issue, Shafia et al. (2011) propose to restate the sequence constraints in such a way that the processing times for all preceding tasks of the same job or processed on the same machine are explicitly considered. Therefore, we will rewrite constraints (3.8) as well as (3.9). For the former constraint this can be done in a quite straight forward fashion.

$$C_{j,l} \geq C_{j,l'} + \sum_{l''=\ l'+1}^{l} p_{j,l''}, \qquad \forall j \in J,\ l \in \{2, \ldots, l_j\},\ l' < l \tag{4.14}$$

These new constraints can be interpreted as follows. In the case of three tasks, i.e. $l_j = 3$, the constraint ensures that

$[l = 2,\ l' = 1]$: task 2 cannot end before task 1 is completed and the processing time for task 2 has passed,

$[l = 3,\ l' = 2]$: task 3 cannot end before task 2 is completed and the processing time for task 3 has passed,

$[l = 3,\ l' = 1]$: task 3 cannot end before task 1 is completed and the processing times for task 2 and task 3 have passed.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

The last constraint is the one that has been added to the original problem and its interpretation shows how the dependencies of a task's completion time on all previous processing times are now embedded into the model constraints.

With the analogous considerations the second sequence constraint, constraint (3.9), is supposed to be transformed as well. This cannot be established as simple as for the other one, because in order to add the processing times to the constraint, we first need a way of finding all tasks (not necessarily from the same job) which proceed the task (j, l) considered on the left-hand side of inequality (3.9) on machine i.

In order to do this, we will introduce two new binary auxiliary variables, $\phi_{j,l,h,z}$ and $\rho_{j,l,h,z,f,g,i}$. The variable $\phi_{j,l,h,z}$ is set to one if the task (j, l) is completed after (h, z). Thus, we have to add the following constraints in order to set this variable correctly.

$$\forall j \in J_0,\ l = 1, \dots, l_j,\ h \in J_0,\ z = 1, \dots, z_h :$$

$$C_{h,z} - C_{j,l} \geq -L \cdot \phi_{j,l,h,z} \tag{4.15}$$

$$C_{j,l} - C_{h,z} \geq -L(1 - \phi_{j,l,h,z}) \tag{4.16}$$

$$\phi_{j,l,h,z} + \phi_{h,z,j,l} \geq 11 \tag{4.17}$$

Thus, if (j, l) is scheduled after (h, z) the left-hand side of the first inequality will be negative and $\phi_{j,l,h,z}$ is forced to be one and the second constraint will be fulfilled automatically. If it is the other way round, the first constraint will be fulfilled anyway whereas the second constraint will force $\phi_{j,l,h,z}$ to be zero. The last constraint is necessary to set at least one value of $\phi_{j,l,h,z}$ or $\phi_{h,z,j,l}$ to one if $C_{j,l} = C_{h,z}$.

We also have to set the value for the variable $\rho_{j,l,h,z,f,g,i}$. It should be 1 if, and only if, the operation (f, g) is a (not necessarily direct) successor of (h, z) and a predecessor of (j, l) on machine i. This means that (f, g) has to fulfill, that

- it is actually scheduled on machine i,
- it is completed after (h, z) and
- it is completed before (j, l).

To assign the correct values to ρ, we have to add the following constraints.

$$\forall j \in J,\ l = 1, \dots,\ l_j,\ h \in J_0,\ z = 1, \dots, h_z,\ f \in J,\ g = 1, \dots, l_f,\ i \in M :$$

$$L\rho_{j,l,h,z,f,g,i} \geq \sum_{j' \in J_0} \sum_{l'=1}^{l_{j'}} x_{f,g,j',l',i} + \phi_{f,g,h,z} + \phi_{j,l,f,g} - 2 \tag{4.18}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

$$-L(1-\rho_{j,l,h,z,f,g,i}) \leq \sum_{j' \in J_0} \sum_{l'=1}^{l_{j'}} x_{f,g,j',l',i} + \phi_{f,g,h,z} + \phi_{j,l,f,g} - 3 \tag{4.19}$$

The double sum on the right-hand side only takes the value one if the operation (f, g) has a predecessor on machine i, which due to including the dummy job is always the case if (f, g) is actually scheduled on machine i. Constraint (4.18) ensures that $\rho_{j,l,h,z,f,g,i}$ has to take the value 1 in case all three terms on the right-hand side have the value one. The second constraint (4.19), on the other hand, forces $\rho_{j,l,h,z,f,g,i}$ to be zero in case at least one of the terms is zero.

Having established the values for the auxiliary variables, the new sequence constraint can be formulated as follows.

$$\begin{aligned}
&\forall j \in J,\ l = 1, \ldots,\ l_j,\ h \in J_0,\ z = 1, \ldots, h_z,\ i \in M: \\
&C_{j,l} \geq C_{h,z} + \sum_{\substack{j' \in J,\ l'=1,\ldots,l_{j'}, \\ (j',l') \neq (h,z)}} \rho_{j,l,h,z,j',l',i} \cdot p_{j',l'} \\
&\qquad - L(1-\rho_{j,l,0,1,h,z,i}) - L(1 - \sum_{j' \in J_0} \sum_{l'=1}^{l_{j'}} x_{j,l,j',l',i})
\end{aligned} \tag{4.20}$$

Since every operation is completed after the dummy job, if $\rho_{j,l,0,1,h,z,i} = 0$ holds, the constraint becomes irrelevant because the operation (h, z) is either completed after operation (j, l) or it is not scheduled on machine i. Since all machines can only operate one operation of one job at a time, the first case means that (h, z) is scheduled after (j, l). In any case, (h, z) can never be a predecessor of (j, l) on the same machine if $\rho_{j,l,0,1,h,z,i} = 0$.

On the other hand, if $\rho_{j,l,0,1,h,z,i} = 1$, the operation (h, z) is scheduled before (j, l) on machine i and thus, the completion time of the posterior operation has to be greater or equal to the completion time of the prior operation plus the processing times for all operations scheduled on machine i in between those operations.

This will only ensure that the operation (h, z) is scheduled prior to (j, l) on machine i. However, we still have to make sure that (j, l) is even scheduled on machine i. This can be done by including the term $L(1 - \sum_{j' \in J_0} \sum_{l'=1}^{l_{j'}} x_{j,l,j',l',i})$, such that the constraint will be fulfilled trivially in case that $\sum_{j' \in J_0} \sum_{l'=1}^{l_{j'}} x_{j,l,j',l',i} = 0$, i.e. (j, l) is not scheduled on machine i.

In summary, constraint (4.20) is only relevant for combinations of operations (j, l) and (h, z) with machine i, for which both operations are indeed scheduled on i and (h, z) is scheduled before operation (j, l).

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

4.7 Robust model 2

The main idea behind this robust approach is to manage the trade-off between a solution that is as close to the optimal value as possible but stays feasible for a great range of potential values the uncertain parameters might attain. It aims at avoiding an excessive degree of conservatism, which results in focusing overly on the robustness of the solution and missing out on optimality (Bertsimas and Sim (2004)). In Bertsimas and Sim (2004) this phenomenon is called the "price of robustness". In the following, the ideas behind the approach are presented, which are taken from Bertsimas and Sim (2004).

If we consider a linear inequality constraint i such as $\sum_j a_{ij}x_j \leq b_i$, where a_i is a random coefficient vector, although all elements a_{ij}, with j being an element of the index set J_i, are uncertain, in reality not for all $j \in J_i$ the coefficients a_{ij} will turn out to be subject to perturbation. Thus, we can define a number $\Gamma_i \in [0, |J_i|]$, which does not have to be an integer, for each constraint i. It captures how many coefficients are assumed to deviate from their nominal values. A robust schedule is then constructed under the assumption that for each constraint i not more than Γ_i coefficients will be uncertain. Thus, the solution to this model will be feasible independently of the values the uncertain coefficients attain, as long as only $\lfloor \Gamma_i \rfloor$ coefficients vary. We present an outline of the way this is included into a model.

For every constraint a so-called protection function is added, assuming that the positive random coefficients $\tilde{a}_{ij}$ are symmetrically distributed around the nominal value a_{ij} and take values within the interval $[a_{ij} - \hat{a}_{ij},\ a_{ij} + \hat{a}_{ij}]$.

$$\max_{\{S_i \cup \{t_i\},\ S_i \subseteq J_i,\ |S_i| = \lfloor \Gamma_i \rfloor,\ t_i \in J_i \setminus S_i\}} \left\{ \sum_{j \in S_i} \hat{a}_{ij}|x_j| + (\Gamma_i - \lfloor \Gamma_i \rfloor)\ \hat{a}_{it_i}|x_j| \right\} \tag{4.21}$$

This protection function (4.21) can be interpreted as the largest value of deviations from the nominal values considering all scenarios in which for a subset $S_i \subseteq J_i$ with exactly $\lfloor \Gamma_i \rfloor$ elements the coefficients attain their worst case values and one coefficient with index t_i will attain $(\Gamma_i - \lfloor \Gamma_i \rfloor) \cdot 100$ percent of the worst case value. It is important to note that in the case of $\Gamma_i = 0$, the problem is the original one with the assumption of no uncertainty in the data. In the other border case, $\Gamma_i = |J_i|$, we obtain the complete conservative approach. This shows how the choice of Γ_i is able to control the degree of robustness.

The protection function (4.21) is the core of the approach and the equivalent representation of (4.21) as an optimization problem can be used to incorporate it into the original optimization problem to obtain the robust version.

A very important additional property of a robust model constructed with the approach

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

above, which is proven in Bertsimas and Sim (2004), is that the solution to the resulting model is not only feasible in any case for which at most $\lfloor \Gamma_i \rfloor$ coefficients vary, but also in the case that more coefficients are subject to perturbation the solution stays feasible with a high probability.

4.7.1 Application to the problem

In Lin et al. (2004) this approach is directly applied to the corresponding sequence constraints, for which the structure of the constraints resemble the ones given in Bertsimas and Sim (2004).

We also want to develop robust model in the case of our problem at hand. Since the processing times are discrete, strictly speaking the approach cannot be applied to our model without hesitation, because it would require continuous values. Nevertheless, we still want to apply the approach to our deterministic model using the newly transformed constraints (4.14) and (4.20). Without any information about the approval probabilities of the jobs, we will assume that the nominal value of each uncertain processing time is zero. This means that the batches will all be approved. The maximum deviation from this nominal value is given by $p_{j,O_{j,l}}$. We will denote the index set of uncertain processing times P as above.

We have to think about the way the degree of robustness Γ should be defined for our problem. It would not be reasonable to consider every processing time of the same job individually. This is because of two reasons.

- If a batch gets rejected, both operations, "mixing" and "testing", have to be repeated. Thus, it will not make sense to assume that the processing time of the rework mixing operation might vary, but the corresponding testing operation not.
- If we assume that a batch can be rejected several times, it will not be reasonable to assume that the n-th rework processing times are not subject to perturbations, but the $(n+1)$-th again are.

Therefore, we will consider a global Γ, which means that it is equal for all constraints. The value of $\lfloor \Gamma \rfloor$ will represent the number of jobs, for which it is assumed that the processing times might be subject of perturbations, and the processing times of one job can vary by $(\Gamma - \lfloor \Gamma \rfloor) \cdot 100$ percent.

Sequence Constraint 1

We will first take a look at the protection function for the sequence constraint (4.14). Each inequality only contains processing times of one job $j \in J$. Since we regard the

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

processing times of one and the same job as one block of uncertain parameters, there is no real protection function but instead for $\Gamma > 0$ we have to consider the worst case values of the stochastic processing times appearing in the constraints.

$$C_{j,l} \geq C_{j,l'} + \sum_{l''=\ l'+1}^{l} p_{j,O_{j,l''}}, \qquad \forall j \in J,\ l \in \{2,\dots,l_j\},\ l' < l \tag{4.22}$$

Thus, the robust approach does not bring any transformation to the constraint (4.14), but compared to the robust model in chapter 4.5, among operations of the same jobs we are now considering the worst case of all processing times in between the operations.

Sequence Constraint 2

We consider now the second sequence constraint which we have transformed in such a way, that for each combination of operations (j,l) and (h,z) all processing times in between appear explicitly in the inequalities. We will denote the set P as above, as an index set of all operations with stochastic processing times. With this set we can rewrite the constraint (4.20) by splitting the sum in a stochastic and a deterministic part.

$$\begin{aligned}
&\forall j \in J,\ l = 1,\dots,\ l_j,\ h \in J,\ z = 1,\dots,h_z,\ i \in M: \\
&C_{j,l} \geq C_{h,z} + \sum_{\substack{(j',l')\notin P,\\(j',l')\neq(h,z)}} \rho_{j,l,h,z,j',l',i} \cdot p_{j',l'} + \sum_{\substack{(j',l')\in P,\\(j',l')\neq(h,z)}} \rho_{j,l,h,z,j',l',i} \cdot p_{j',l'} \\
&\quad - L(1-\rho_{j,l,0,1,h,z,i}) - L\left(1 - \sum_{j'\in J_0}\sum_{l'=1}^{l_{j'}} x_{j,l,j',l',i}\right)
\end{aligned} \tag{4.23}$$

All processing times of the first sum are deterministic, hence there is no need to consider this part any further. The processing times appearing in the second sum, on the other hand, are all stochastic. Thus, we will apply the robust approach and add a protection function to the inequality. Note that the nominal values of processing times for operations in the set P are zero. This is why we can disregard the second sum and only add the protection function instead. We obtain the following new constraint.

$$\begin{aligned}
&\forall j \in J,\ l = 1,\dots,\ l_j,\ h \in J,\ z = 1,\dots,h_z,\ i \in M: \\
&C_{j,l} \geq C_{h,z} + \sum_{\substack{(j',l')\notin P,\\(j',l')\neq(h,z)}} \rho_{j,l,h,z,j',l',i} \cdot p_{j',l'} - L(1-\rho_{j,l,0,1,h,z,i}) - L\left(1 - \sum_{j'\in J_0}\sum_{l'=1}^{l_{j'}} x_{j,l,j',l',i}\right)
\end{aligned}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

$$+ \underbrace{\max_{\substack{S\cup\{t\},\ S\subseteq J,\\ |S|=\lfloor\Gamma\rfloor,\ t\in J\setminus S}} \left\{ \sum_{j'\in S} \sum_{\substack{(j',l')\in P,\\ (j',l')\neq(h,z)}} \rho_{j,l,h,z,j',l',i} \cdot p_{j',O_{j',l'}} + (\Gamma - \lfloor\Gamma\rfloor) \sum_{\substack{(t,l')\in P,\\ (t,l')\neq(h,z)}} \rho_{j,l,h,z,t,l',i} \cdot p_{t,O_{t,l'}} \right\}}_{\text{protection function}} \tag{4.24}$$

The protection function evaluates the largest additional processing time that needs to be considered in the case that $|S| = \lfloor\Gamma\rfloor$ jobs have uncertain processing times and one job's processing times vary with a rate of $(\Gamma - \lfloor\Gamma\rfloor)$.

In order to include constraint (4.24) into our model, we have to rewrite the *max*-statement as the objective function of a linear optimization problem. This will be done according to Proposition 1 in Bertsimas and Sim (2004).

$$\max \sum_{j'\in S} \left[e_{j,l,h,z,i,j'} \sum_{\substack{(j',l')\in P,\\ (j',l')\neq(h,z)}} (\rho_{j,l,h,z,j',l',i} \cdot p_{j',O_{j',l'}}) \right] \tag{4.25}$$

$$\text{s.t.} \quad \sum_{j'\in J} e_{j,l,h,z,i,j'} \leq \Gamma \tag{4.26}$$

$$0 \leq e_{j,l,h,z,j'} \leq 1, \qquad \forall j' \in J \tag{4.27}$$

The optimal values of the variables $e_{j,l,h,z,i,j'},\ j' \in J$ will be set for each constraint given by (j, l, h, z, i) such that all processing times of exactly $\lfloor\Gamma\rfloor$ jobs attain their worst case values.

Following the proof for Theorem 1 in Bertsimas and Sim (2004), the dual problem is given by

$$\min \sum_{j'\in J} \eta_{j,l,h,z,i,j'} + \Gamma \cdot e_{j,l,h,z,i} \tag{4.28}$$

$$\text{s.t.} \quad e_{j,l,h,z,i} + \eta_{j,l,h,z,i,j'} \geq \sum_{\substack{(j',l')\in P,\\ (j',l')\neq(h,z)}} \rho_{j,l,h,z,j',l',i} \cdot p_{j',O_{j',l'}}, \qquad \forall j' \in J \tag{4.29}$$

$$\eta_{j,l,h,z,i,j'} \geq 0, \qquad \forall j' \in J \tag{4.30}$$

$$e_{j,l,h,z,i} \geq 0, \qquad \forall j \in J,\ l = 1,\dots,l_j,\ h \in J,\ z = 1,\dots,h_z,\ i \in M \tag{4.31}$$

According to Theorem 1 in Bertsimas and Sim (2004), we obtain a robust model formulation of the deterministic model by removing the sequence constraints (3.8) and (3.9) from the model presented in chapter 3.2.3 and adding the constraints (4.29)-(4.31) obtained by the dual problem. The protection function of the transformed sequence constraint (4.20)

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

is replaced by the objective function of the dual model. In summary, we add the following constraints to the deterministic model.

$$C_{j,l} \geq C_{h,z} + \sum_{\substack{(j',l') \notin P, \\ (j',l') \neq (h,z)}} \rho_{j,l,h,z,j',l',i} \cdot p_{j',l'} + \sum_{j' \in J} \eta_{j,l,h,z,i,j'} + \Gamma \cdot e_{j,l,h,z,i}$$

$$- L(1 - \rho_{j,l,0,1,h,z,i}) - L \left(1 - \sum_{j' \in J_0} \sum_{l'=1}^{l_{j'}} x_{j,l,j',l',i} \right) \tag{4.32}$$

$$e_{j,l,h,z,i} + \eta_{j,l,h,z,i,j'} \geq \sum_{\substack{(j',l') \in P, \\ (j',l') \neq (h,z)}} \rho_{j,l,h,z,j',l',i} \cdot p_{j',O_{j',l'}}, \qquad \forall j' \in J \tag{4.33}$$

$$\eta_{j,l,h,z,i,j'} \geq 0, \qquad \forall j' \in J \tag{4.34}$$

$$e_{j,l,h,z,i} \geq 0, \qquad \forall j \in J,\ l = 1, \ldots, l_j,\ h \in J,\ z = 1, \ldots, h_z,\ i \in M \tag{4.35}$$

Of course, also the constraints for the values of the auxiliary variables (4.15)-(4.19) have to be added to the deterministic model.

4.7.2 Implementation in MATLAB

We implemented the robust counterpart of the deterministic model with transformed sequence constraints. This has been accomplished by the function *detModel_robust* in MATLAB (MATLAB (2019)), for which the complete source code is provided in the appendix. The function is based on the deterministic model and has been adapted to the robust formulation by adjusting the handling of parameter values such as the processing times, adding the new auxiliary variables and replacing the second sequence constraint by constraints (4.32)-(4.35).

The solution to the robust model provides us with the following. We obtain a schedule that considers the point in time at which the operations start and, more importantly, what operations will be processed on which machine and in which order. Thus, we can use this order to start the process. After each operation is done, we know exactly which operation succeeds. As a result, we can start the next operation on each machine as soon as possible, i.e. as soon as the job is ready for this operation. Thus, in the case of 3 jobs, which will only be rejected at most once, there are 8 possible scenarios for which we obtain 8 different schedules. All of these schedules have the same allocation of operations to machines and order of processing on each machine but the end times of the operations vary. Also, the resulting make spans will depend on the scenario that realized in the end.

In order to find the actual realization of a schedule for a given allocation, we have created the function *detModel_withStartingSolution*, which takes the allocation of the schedule and

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

the actual processing times, that determine the scenario, as an input. It then calculates the schedule that actually realizes given by the end times of the operations. This function can therefore be used to evaluate the robustness of a schedule by calculating the make spans for different scenarios.

It is important to repeat that we chose to use the worst case assumption for the sequence constraints within one job and to use the robust constraints developed above for the sequence of operations on each machine. In the original approach, $\Gamma = 1$ meant that if at most one job gets rejected, the operation plan still stayed feasible. This has to hold for any job, and therefore in our problem, the schedule will be build in such a way that it will be feasible no matter how many jobs will actually be rejected, since the model has to consider each rework of every job. This is due to the fact, that the processing times cannot vary within a continuous interval but instead either the full processing time is needed or none. Hence, the value of Γ cannot be used to steer the degree of the robustness in this application to our problem. However, we will see in the example below, that the approach is able to find a robust schedule which at least performs better than the robust schedule obtained by the deterministic model. In order to show how interpret and measure the robustness of the obtained schedule we will present a small example.

Example 4.2

We begin at the starting point of **Example 4.1** once again and assume the same setting and processing times. First, we have to choose a value for Γ. Since we have 3 jobs, it has to hold that $\Gamma \in [0, 3]$. With the value of Γ and the other parameters' setting we can use the function *detmodel_robust* to find a solution to the problem. For example, choosing $\Gamma = 1$ results in the following schedule.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

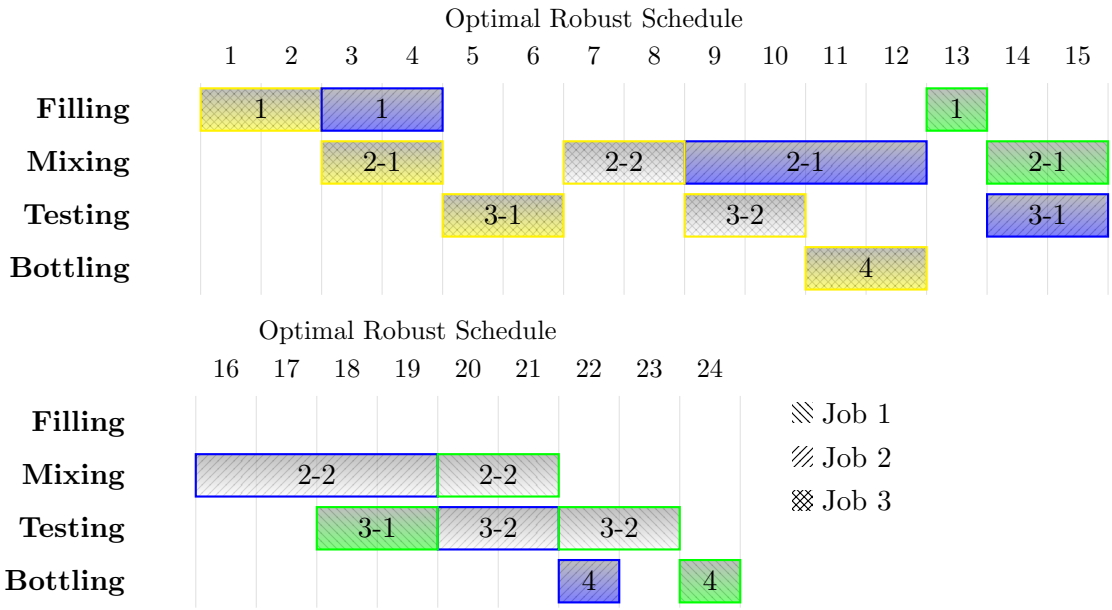

Figure 4.3: The robust schedule for **Example 4.2** with $\Gamma = 1$, own drawing

The reworking operations are colored in grey in order to emphasize that whether or not they will actually take place depends on the outcomes of the quality control. It can be seen that all rework operations are scheduled properly such that no matter which scenario realizes the schedule stays feasible. The maximum number of time periods it takes in any scenario is 24. We can now compare this to the schedule we obtain from the robust model 1 in chapter 4.5, which is equal to calculating the worst case scenario schedule using the function *detModel*. For the comparison, we will consider each of the 8 scenarios and use the function *detModel_withStartingSolution* to calculate the make spans that will actually realize in each case.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

job(s) being rejected	schedule 4.3	worst case det. schedule
none	15	16
1	18	20
2	20	19
3	19	20
1 & 2	23	23
1 & 3	22	24
2 & 3	21	20
1 & 2 & 3	24	24

Table 4.2: Make span comparison of robust and worst case deterministic schedules for all possible scenarios

This table shows, that both schedules perform equally in the worst case with a make span of 24, but since the robust schedule also aimed at optimizing the value in each scenario, it outperforms the worst case schedule in many cases, such as jobs 1 and 3 being rejected. We can compare the mean value of the worst case solution, $\mu_{\text{worst case}} = 20.75$, and the standard deviation, measured as above with formula (4.1), $\sigma_{\text{worst case}} = 2.7646$, to the mean, $\mu_{\text{robust}} = 20.25$, and standard deviation, $\sigma_{\text{robust}} = 2.9155$, of the robust schedule in figure 4.3. Since it holds $\mu_{\text{worst case}} < \mu_{\text{robust}}$, the robust schedule on average has a smaller make span. Although it holds that $\sigma_{\text{robust}} > \sigma_{\text{worst case}}$ in absolute values, relative to the make spans, the robust schedule performs better than the one obtained from the deterministic model. In conclusion, both schedules are robust, i.e. they are feasible in all scenarios, but the robust schedule is more solution robust, which means that on average the make spans stay closer to the optimum.

4.7.3 Remarks and summary of the approach

The transformation of the sequence constraints as well as the robust model restrictions entails a great number of additional constraints compared to the original deterministic model. This makes it such a computational challenge, that even for the scheduling of 4 jobs it already takes hours to complete.
To sum up the results of the approach, it can be said that the idea presented in Shafia et al. (2011) for scheduling with uncertain processing times cannot be applied purposefully to our setup in the sense that the degree of robustness can be adjusted. However, the approach still provides a schedule which is model as well as solution robust.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

5 Stochastic Model

We will now present a stochastic approach to the problem at hand which is introduced by an example.

Example 4.1 (continued)

To provide another incentive for including the stochastic character of the quality control into the model, we start again with **Example 4.1** presented in the previous chapter. Now, we assume that job 3 is not accepted the first time it goes through the quality control. This means, that it has to repeat operations 2 and 3. Starting the production in the order according to the schedule above, this would result in the following realization of the production plan.

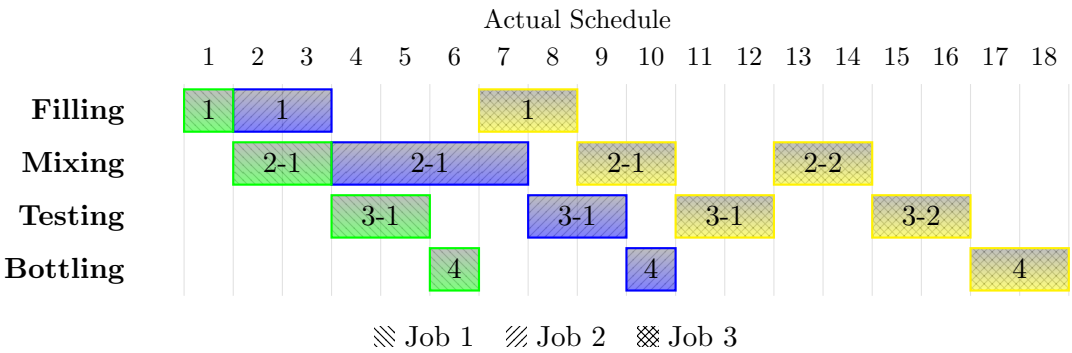

Figure 5.1: The realization of the schedule for **Example 4.1** in case job 3 gets rejected once, own drawing

This leads to a schedule with the actual make span of 18. Since job 3 was scheduled as the last job to start operating and thus has no waiting times for any machine, the new make span is the old one plus the sum of the processing times for operations "Mixing" and "Testing": $C^{\max} = 14 + (2+2) = 18$.

It would have been possible to achieve a lower make span if the order in which the jobs started was chosen differently. We now assume that we had known in advance that job 3

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

would need rework and thus that its operation sequence expands to $[1, 2, 3, 2, 3, 4]$, while the sequences for the other jobs stay the same. Solving the deterministic model again with these parameters yields the following schedule with minimal make span of 16 time periods.

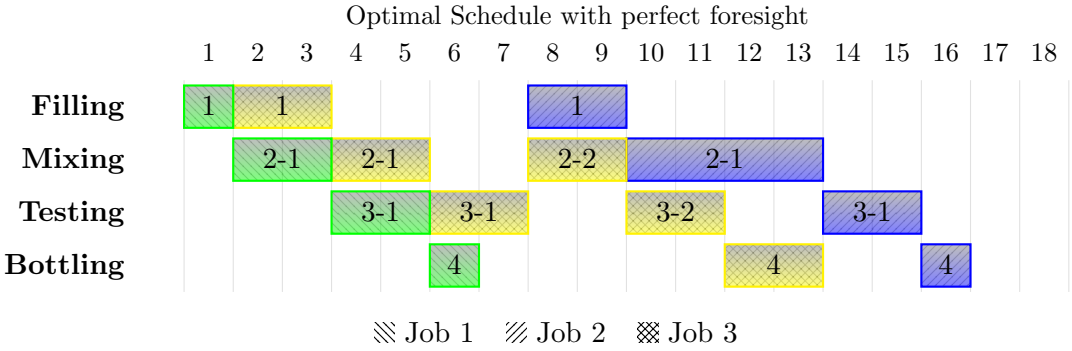

Figure 5.2: The optimal schedule with perfect foresight for **Example 4.1** in case job 3 gets rejected once, own drawing

In this schedule the order in which the jobs start operating is changed to optimally allocate the operations. Knowing that job 3 needs a rework leads to starting it as the second job instead of job 2.

Of course, we cannot assume that we will actually have perfect foresight, otherwise the parameters would not be uncertain anymore, but it still shows that it might be advantageous to take into account the stochastic characteristic of the production process because otherwise we would risk a very bad realization of the schedule when a batch gets rejected unexpectedly.

In summary, the examples above give reasons to explicitly include the stochastic characteristic of the quality control into the model with the aim of providing realizations that still result in make spans as close to the optimal in case of perfect foresight as possible. This will be done in the following chapters. First, we will familiarize with important concepts of stochastic programming, which will be needed throughout the rest of this chapter. Then, we will start with the stochastic approach to the problem at hand.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

5.1 Introduction to stochastic programming

This introduction to concepts of stochastic programming is based on Ahmed (2017).
A typical deterministic optimization problem can be summarized in the following way.

$$\min_x \{F(x,\xi) : x \in X(\xi) \subseteq \mathbb{R}^n\}, \tag{5.1}$$

with x being the decision vector, $F : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ the objective function and X a set of constraints. This constraint set, for instance, has the form $X(\xi) := \{x : G(x,\xi) \leq 0\}$ with a function $G : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^m$ representing the constraints. It is important to notice that in this case, both, F and X, depend on a vector of parameters $\xi \in \mathbb{R}^d$. In many real world problems the actual values of these parameters are not given at the time the optimization problem has to be solved. One reason for this might be that the parameters depend on information that can only be given in the future. This leads to the idea of *Stochastic Optimization*, which treats the parameters with uncertain values as random variables, denoted by $\tilde{\xi}$. Typically, the assumption holds that the distribution of $\tilde{\xi}$ is known and does not depend on the decision variables. Since the values of F and X are influenced by the realization of the parameters, it is necessary to define optimality as well as feasibility in this new setting of uncertainty. To present the ideas behind the corresponding new concepts of these two terms, we will consider the case of an objective function depending on uncertain parameters and the case of a constraint set depending on uncertain parameters separately.

5.1.1 Objective function depending on random variables

We assume that the objective function depends on the random variables wrapped up in the random parameter vector $\tilde{\xi}$. The constraint set, on the other hand, is deterministic. This means that only after the realization of the parameters we are able to assign an actual value to the objective function. Since the decision about the values of x are made before the parameters attain their actual values, we need to come up with a new way of deciding what makes a value of the objective function optimal. The most common approach is to choose the values of the decision variables x in such a way, that the **expected** value of the objective function is minimized. Hence, we obtain the following stochastic version of the deterministic problem (5.1):

$$\min_x \left\{\mathbb{E}[F(x,\tilde{\xi})] : x \in X \subseteq \mathbb{R}^n\right\}. \tag{5.2}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Example: newsvendor problem

One of the most famous examples of stochastic optimization problems is the so-called newsvendor problem. As described in (Birge and Louveaux, 2011, Chapter 1.1 e), a newsvendor boy has to decide how many newspapers he buys in the morning, without knowing how many he will be able to sell. He cannot reuse newspapers for the next day and when he returns the unsold ones, he gets a refund r smaller than the price w he paid in the morning. If we denote the random variable of the number of papers sold for a price q in a day by $\tilde{\xi}$, the objective function takes the following form

$$\min_x cx + \mathcal{D}(x, \tilde{\xi}) = cx + \mathbb{E}_\xi \left[-q \min(\tilde{\xi}, x) - r \max(x - \tilde{\xi}, 0) \right], \tag{5.3}$$

where x is the decision variable denoting the numbers of papers bought in the morning, cx is the deterministic part, because these costs only depend on the value of x, and $\mathcal{D}$ is the stochastic part, depending on the actual value of $\tilde{\xi}$, the number of newspapers sold. This is one of few examples, where in fact an analytical solution exists, depending on the distribution function of $\tilde{\xi}$.

5.1.2 Chance constraint stochastic optimization

In the second case, the objective function is independent of the uncertainty, whereas the constraint set depends on random variables. The idea for defining "feasibility" is to specify a significance level ϵ and to request that x belongs to the stochastic constraint set with a probability of at least $1 - \epsilon$. This stochastic variant of (5.1) is formally described by

$$\min_x \left\{ F(x) : \mathbb{P}[G(x, \tilde{\xi}) \leq 0] \geq 1 - \epsilon,\ x \in S \right\}, \tag{5.4}$$

where $S \subseteq \mathbb{R}^n$ is the deterministic part of the feasible region.

Example: stochastic demand

A good example for a chance constraint stochastic optimization problem arises in logistics problems, for which the demand is unknown. In the deterministic case, the constraint might be that the amount delivered to a demand point must be at least the amount demanded and the constraint might take the following form.

$$\sum_{i=1}^{n} v_i \geq u, \tag{5.5}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

$v_i \ldots$ amount delivered from supply location i
$u \ldots$ amount demanded at a location

In the stochastic case, when the exact demand is not known beforehand, we can choose a significance level ϵ, for instance $\epsilon = 0.02$, and the constraint transforms to

$$\mathbb{P}\left[\sum_{i=1}^{n} v_i \geq u\right] \geq 0.98. \tag{5.6}$$

Thus, in 98% of the time the demand is actually met.

5.1.3 Extensions

Both cases discussed above belong to the special case where the underlying deterministic model is a linear optimization problem, but there are many other applications of stochastic optimization theory as well. One example is stochastic dynamic programming. It might also be of more interest for the decision maker to use some kind of risk measure instead of the expected value of the objective function. Additionally, it might occur that not even the distribution of the random variables is given with certainty, in which case not only perturbations of the parameters but also possible perturbations of their distribution have to be taken into consideration when solving the optimization problem.

Since our deterministic model belongs to the MILP problems and we assume the distribution to be given, only the cases described above are relevant for further discussions in this thesis and we therefore omit the detailed description of other possible extensions.

5.2 Stochastic model

The introduction above should provide the reader with a small insight into the concepts of stochastic modeling. The **Example 4.1** presented an incentive to take into account a certain variability in the values of the parameters when creating a schedule in order to avoid a deterioration of the actual schedule in case of unforeseen rejections of the quality control.

5.2.1 Stochastic approach

The following approach is based on the one developed in Balasubramanian and Grossmann (2002). Although in their paper it is done for a flow-shop scheduling problem, the approach can be used analogously for the problem at hand. The main idea is to consider the set

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

of all possible scenarios S, where a scenario is defined as a "combination of realizations of all random variables" (Balasubramanian and Grossmann (2002)). This means that one scenario is given by all random variables, where each one attains exactly one value out of its respective value set. Then, for every possible scenario the make span can be calculated. This can be done by simply solving the deterministic model, because each scenario is defined by the actual values of the parameters.

The objective function (5.7), in our problem only consisting of the variable $C^{\max}$, is now a random variable. We minimize the expected value of this random variable regarding all possible scenarios in order to obtain an optimal solution in the stochastic sense. This expected value is given by the weighted sum of each scenario's make span and the weights ω_s correspond to the probabilities with which scenario $s \in S$ occurs.

The allocation variables $x_{j,l,h,z,i}$ are not random variables but determine the operation sequence independently of the scenario. Therefore, the allocation constraints are not altered in the stochastic model. The time variables C and $C^{\max}$, on the other hand, indeed depend on the respective scenario, because the completion times will vary with the actual processing times of each job depending on the outcomes of the quality control according to the scenario. Additionally, the auxiliary variables a, b and w that keep track of the jobs in the system can vary between scenarios, since it might be possible that due to the scenario dependent completion times in one scenario s_1 a job j is bottled earlier than in another scenario such that in the first case the time in the system of this job does not intersect with another job h anymore, whereas in the latter case, in scenario s_2, it does. This results in different values of the variables, e.g. $w_{j,h,s_1} \neq w_{j,h,s_2}$. Since all those variables now depend on the scenario, not only the objective function has to be adapted compared to the deterministic case but the set of constraints is extended as well. All constraints containing at least one of the time variables, auxiliary variables or random parameters p need to be adapted to the effect that they need to hold for any scenario $s \in S$. The complete stochastic model derived from the deterministic model in chapter 3 is given in the following. As already noted, the objective function (5.7) and constraints (5.8), (5.14)-(5.26) as well as the non-negativity constraints (5.29)-(5.31) and (5.33)-(5.34) differ from the deterministic model in such a way that they are considered for each scenario $s \in S$.

$$\min \ \mathbb{E}(C^{\max}) = \sum_{s \in S} \omega_s C_s^{\max} \tag{5.7}$$

subject to

$$C_s^{\max} \geq C_{j,l_j,s}, \qquad \forall j \in J, \ \forall s \in S \tag{5.8}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

$$\sum_{h=0}^{n}\sum_{z=1}^{z_h}\sum_{i=1}^{m} x_{j,l,h,z,i} = 1, \qquad \forall j \in J,\ l = 1,\dots,l_j \tag{5.9}$$

$$\sum_{h=0}^{n}\sum_{z=1}^{z_h} x_{j,l,h,z,i} \leq \alpha_{j,l,i}, \qquad \forall j \in J,\ l = 1,\dots,l_j,\ i \in M \tag{5.10}$$

$$\sum_{j=1}^{n}\sum_{l=1}^{l_j}\sum_{i=1}^{m} x_{j,l,h,z,i} \leq 1, \qquad \forall h \in J,\ z = 1,\dots,z_h \tag{5.11}$$

$$\sum_{j=1}^{n}\sum_{l=1}^{l_j} x_{j,l,0,1,i} \leq 1, \qquad \forall i \in M \tag{5.12}$$

$$\sum_{j=1}^{n}\sum_{l=1}^{l_j} x_{j,l,h,z,i} \leq \sum_{j=0}^{n}\sum_{l=1}^{l_j} x_{h,z,j,l,i}, \qquad \forall h \in J,\ z = 1,\dots,z_h,\ i \in M \tag{5.13}$$

$$C_{j,1,s} \geq p_{j,1,s}, \qquad \forall j \in J,\ \forall s \in S \tag{5.14}$$

$$C_{j,l,s} \geq C_{j,l-1,s} + p_{j,l,s}, \qquad \forall j \in J,\ l = 2,\dots,l_j,\ \forall s \in S \tag{5.15}$$

$$C_{j,l,s} \geq C_{h,z,s} + \sum_{i=1}^{m}(x_{j,l,h,z,i} \cdot set_{j,h,i}) + p_{j,l,s} - L \cdot \left(1 - \sum_{i=1}^{m} x_{j,l,h,z,i}\right), \tag{5.16}$$

$$\forall j \in J,\ l = 1,\dots,l_j,\ h \in J,\ z = 1,\dots z_h,\ \forall s \in S$$

$$\forall j,h \in J,\ j \neq h,\ \forall s \in S:$$

$$(C_{h,1.s} - p_{h,1,s}) - (C_{j,1,s} - p_{j,1,s}) + L \cdot (1 - a_{j,h,s}) \geq 0 \tag{5.17}$$

$$(C_{j,1,s} - p_{j,1,s}) - (C_{h,1,s} - p_{h,1,s}) + L \cdot a_{j,h,s} \geq 0 \tag{5.18}$$

$$a_{j,h,s} + a_{h,j,s} \leq 1 \tag{5.19}$$

$$(C_{j,1,s} - p_{j,1,s}) - C_{h,z_h,s} + L \cdot (1 - b_{j,h,s}) \geq 0 \tag{5.20}$$

$$C_{h,z_h,s} - (C_{j,1,s} - p_{j,1,s}) + L \cdot b_{j,h,s} \geq 0 \tag{5.21}$$

$$\forall j,h \in J,\ j \neq h,\ \forall s \in S:$$

$$w_{j,h,s} - a_{j,h,s} - b_{j,h,s} \leq 0 \tag{5.22}$$

$$a_{j,h,s} - w_{j,h,s} \leq 0 \tag{5.23}$$

$$b_{j,h,s} - w_{j,h,s} \leq 0 \tag{5.24}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

$$\sum_{h \in J, j \neq h} w_{j,h,s} \geq n - v^{\max}, \qquad \forall j \in J,\ s \in S \tag{5.25}$$

$$d_{j,v} + d_{h,v} \leq w_{j,h,s} + w_{h,j,s}, \qquad \forall v \in V,\ j,h \in J,\ j \neq h,\ \forall s \in S \tag{5.26}$$

$$\sum_{v=1}^{v^{\max}} d_{j,v} = 1, \qquad \forall j \in J \tag{5.27}$$

$$x_{j,l,h,z,i} \in \{0,1\}, \qquad \forall i \in M,\ j,h \in J_0,\ l = 1,\dots,l_j,\ z = 1,\dots,z_h \tag{5.28}$$

$$a_{j,h,s} \in \{0,1\}, \qquad \forall j,h \in J,\ s \in S \tag{5.29}$$

$$b_{j,h,s} \in \{0,1\}, \qquad \forall j,h \in J,\ s \in S \tag{5.30}$$

$$w_{j,h,s} \in \{0,1\}, \qquad \forall j,h \in J,\ s \in S \tag{5.31}$$

$$d_{j,v} \in \{0,1\}, \qquad \forall j \in J,\ v = 1,\dots,v^{\max} \tag{5.32}$$

$$C_{j,l,s} \geq 0, \qquad \forall j \in J,\ l = 1,\dots,l_j,\ \forall s \in S \tag{5.33}$$

$$C_s^{\max} \geq 0 \qquad \forall s \in S \tag{5.34}$$

To sum up, the stochastic approach finds a solution, that is feasible for all considered scenarios and results in the smallest expected make span.

Example 5.1

To make the presented stochastic approach more clear, we now apply this model to the setup of **Example 4.1**. In this example the objective was to schedule 3 jobs. We assume that every job has a non-zero probability to be rejected the first time it is tested. For job 1 this probability is $\pi_1 = 0.75$, for job 2 it is given by $\pi_2 = 0.10$ and for job 3 it is $\pi_3 = 0.40$. Thus, job 2 has a very high chance of being approved, whereas for job 1 this only occurs one out of four times. Nevertheless, the jobs are approved the second time they are tested with a probability of 1. In this case the corresponding scenario tree consists of 8 possible scenarios.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

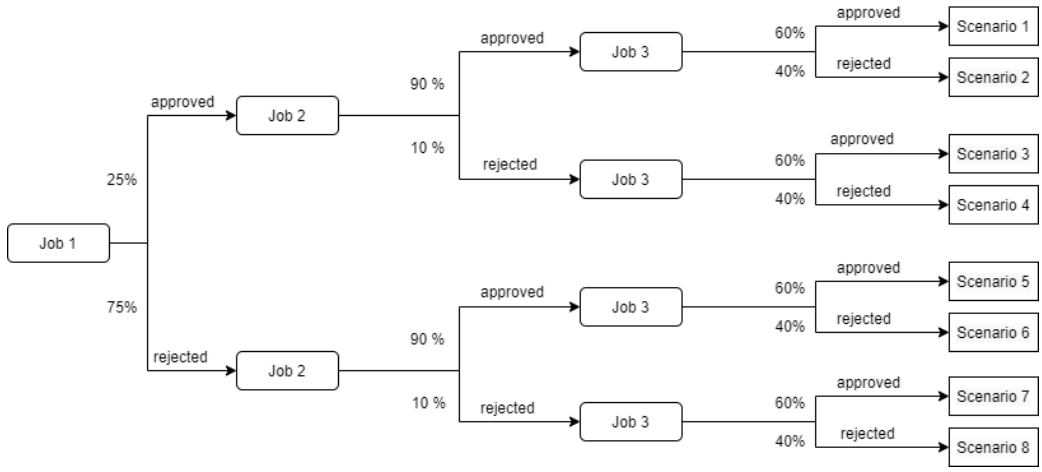

Figure 5.3: Scenario Tree for **Example 5.1**, own drawing

For each scenario we know exactly if a job is rejected the first time or not, so for each scenario we can derive the operation plan according to approvals and rejections. For instance, if we consider scenario Nr. 5, we see that the second and third job are approved the first time they are tested, which means that their operation plans are given by $[1, 2, 3, 4]$. Job 1, on the other hand, is rejected the first time and thus obtains an operation plan in which the operations 2 and 3 are repeated once. Note that this scenario corresponds to the situation considered in chapter 4.1.2.

Solution In order to solve the stochastic approach for this small example, we need to find a solution which is feasible in all of the 8 scenarios, meaning that for each scenario all of the constraints have to hold. Out of these feasible solutions we chose the optimal one by calculating each the expected value of the make spans and choosing the one with the smallest value.

For every feasible allocation the make span can be calculated for each scenario. So, for one allocation we obtain 8 (not necessarily different) make spans. Those make spans are summed up weighted by their probability of occurrence. Since the outcomes of the quality control for different jobs are independent, the probability for each scenario is given by the product of the job's approval or rejection probabilities according to the scenario. For example, scenario 5 has the following probability:

$$\begin{aligned} w_5 &= \mathbb{P}(\text{Job 1 rejected}) \cdot \mathbb{P}(\text{Job 2 approved}) \cdot \mathbb{P}(\text{Job 2 approved}) \\ &= 0.75 \cdot 0.90 \cdot 0.60 \\ &= 0.405. \end{aligned}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Thus, more than one out of three times, the scenario discussed in chapter 4.1.2 will realize. The table below lists the probabilities for each scenario to occur. Note that since we are considering all possible scenarios to evaluate the expected value of the make span, the probabilities sum up to 1.

Scenario	Probability of occurrence
1	0.135
2	0.09
3	0.015
4	0.01
5	0.405
6	0.27
7	0.045
8	0.03
Total	1

Table 5.1: Probabilities of the scenarios in **Example 5.1** according to the scenario tree

It has to be noted that the computation of probabilities might become more complex if the assumption is revoked that the jobs can only be rejected once. In this case conditional probabilities will apply for each job. Nevertheless, the jobs will still be independent among each other.

Implementation in MATLAB To implement the solution procedure of the stochastic model from chapter 5.2.1 in MATLAB (MATLAB (2019)), some changes have to be made to the function *detModel*. The most important difference is to combine the constraints for all scenarios which are considered.

First of all, since we assume that each of the three jobs has a non-zero probability of being rejected the first time, we assign the operation sequence $[1, 2, 3, 2, 3, 4]$ to every job. The different scenarios are represented by the different values of the processing times p. In the deterministic case, it was sufficient to provide the processing times for each task once, assuming that it would not depend on the number of tries the batch already had. In the stochastic case, on the other hand, we need to expand this list of processing times. We need to differentiate between the first time the job goes through the tasks "Mixing" and "Testing", where the processing times are deterministic and given as before in table 4.1, and the second time the job has to go through these tasks. If a job is rejected according to a scenario and actually has to repeat the tasks, the processing times are the same as the first time. If the job has been approved, the processing times are set to zero, which corresponds to the job skipping these tasks. Each scenario has its own individual processing times table.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

Below are the tables for scenarios 2 and 4 as an example. Note that in both scenarios jobs 1 and 3 are approved the first time. The only difference occurs in the processing times of job 2, which are highlighted by a circle.

Operation	1	2-1	3-1	2-2	3-2	4
Job 1	1	2	2	0	0	1
Job 2	2	4	2	0	0	1
Job 3	2	2	2	0	0	2

Table 5.2: Processing times in scenario 2, where job 2 is approved the first time

Operation	1	2-1	3-1	2-2	3-2	4
Job 1	1	2	2	0	0	1
Job 2	2	4	2	4	2	1
Job 3	2	2	2	0	0	2

Table 5.3: Processing times in scenario 4, where job 2 is rejected the first time

The implementation of the stochastic model in MATLAB (MATLAB (2019)) is the function *stochModel* which is based on the function for the deterministic model, but the constraints now loop over the different scenarios in order to have a huge pool of constraints. The input vector scenario has been removed and instead the vector prob has been added, which contains the probability of occurrence for each scenario. The number of scenarios is equal to the length of the probability vector. The objective function has been replaced by the weighted sum of the make spans. The result of this optimization problem, again solved using the IBM CPLEX 12.10 solver (IBM (2019)), is a detailed schedule for each scenario, where the order in which the jobs enter the system and the vessel allocations are the same but the ending times of the operations differ between the scenarios. The complete source code for the function *stochModel* is provided in the appendix.

Using the probabilities calculated above for each scenario yields an optimal schedule in the stochastic sense, in which job 1 enters the system first, followed by job 3. Job 2 starts processing once a vessel is available again. Comparing this result to the one in example chapter 4.1.2 shows that the additional information about the rejection probabilities is considered and the model provides a solution in which the (very likely) rejection of job 1 disrupts the schedule as little as possible, which has not been considered in the best case solution of the deterministic model. For example, the solution of the stochastic model results in a make span for the best case scenario of 15. This can be interpreted as the stochastic allocation taking the loss of 1 time period compared to the optimal schedule from chapter 4.1, since it considers the small probability that this scenario will actually

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

realize and adapting the solution to more likely scenarios instead.

The make spans vary depending on the scenario from 15 in the best case (all are approved) to 25 in the worst cases (scenarios 7 and 8), which yields an expected make span of 18.7. Assuming that we had perfect foresight, such that every scenario occurs with a probability as given in table 5.1 but before the schedule is calculated we would know which scenario will realize and we could use the optimal solution provided by *detModel*. This situation with a priori information is also called *wait-and-see solution* (Birge and Louveaux (2011)). The expected make span for the wait-and-see situation has the value 17.42. The difference in expected make spans of $18.7 - 17.42 = 1.28$ time periods between the stochastic and the wait-and-see solution is referred to as the *expected value of perfect information* (Birge and Louveaux (2011)). It represents how many time periods are additionally needed on average due to the presence of uncertainty. Looking at it from a different perspective, it reveals how much it would be worth enhancing the knowledge about the uncertain elements to get as close to perfect foresight as possible. This might for instance be done by investing in further research.

We can compare the stochastic solution to the deterministic schedule in the worst case, i.e. with all jobs being rejected once, which is feasible for all possible events. In chapter 4.7.2, we have calculated the resulting make span of the latter schedule in each scenario. Assuming that these scenarios occur with the probabilities given above, the expected make span is given by 20.78. This means that the expected make span we obtain from the stochastic approach undercuts the one from the deterministic worst case schedule by $20.78 - 18.7 = 2.08$ time periods, which is a notable improvement. In the same manner, we can compare the stochastic schedule to the robust schedule determined for this example in chapter 4.7 and depicted in figure 4.3. Using the make spans for the different scenarios given in chapter 4.7.2 for the robust schedule, we obtain an expected make span of 19.23 which exceeds the one for the stochastic approach at least by half a time period. Thus, for this example with the assumed rejection probabilities the solution of the stochastic model performs better than the ones provided by the deterministic and robust model.

Problems of this approach

The approach above is quite simple as well as intuitive. Nevertheless, it is easy to see that it is not suitable for any problem of realistic size. This is because the number of scenarios increases exponentially with the number of random variables. For example, if we assume that each job out of n, only needs at most two tries, we obtain two random variables per job. Since either both operations, 2 and 3, are needed or none, we obtain 2^n possible scenarios. Moreover, the already large number of variables and constraints that need to be taken into

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

account by the optimization increases with every additional scenario. One can imagine that this approach quite fast becomes an almost impossible computational challenge with an increasing number of jobs. Thus, the presented stochastic approach renders completely useless in practical applications where hundreds of jobs need to be scheduled.

Since one of the main reasons for this conclusion is the mere number of scenarios, it is vital to find a way of reducing the number of scenarios considered in each optimization. One idea of doing so is given in the next chapter.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6 Reactive scheduling

The presence of uncertainty in the process might cause the actual realization of the schedule to deviate considerably from the original one. Examples of these effects are shown in Cott and Macchietto (1989) and were also presented in **Example 4.1** in chapters 4 and 5 based on our production problem. It shows that one change in the parameters, in this case the processing times, can completely derail the schedule and result in long waiting times on the one hand as well as idle machines on the other hand. In the problem that we are discussing, the parameters do not attain their values all at once or when the production is done, but instead the outcomes of the quality control are known one by one according to the schedule of the quality controls. This gives rise to the following idea. After an initial schedule is calculated, the ongoing process is constantly observed. Once deviations from the original schedule occur, which means that a quality control's outcome is different than assumed for the calculation of the initial schedule, a reacting action is taken. This action can be understood in terms of rescheduling future operations appropriately to adapt to the new situation. In order for this kind of solution approach to have a good performance, it is vital to choose a suitable starting schedule for one but also a useful rescheduling algorithm.

We will first present the general idea behind the rescheduling approach. Afterwards it will be applied to a small example to enhance the understanding of how the algorithm works. As a conclusion to the chapter, possible extensions will be presented.

6.1 Model predictive approach

The approach that we present is a way of producing an optimal production plan by reacting to disturbances in the schedule due to the realization of random parameters. This is applied for instance in production scheduling problems where the risk of machine breakdowns, uncertain processing times or demands are taken into account explicitly in the scheduling process (Ierapetritou and Li (2009)). It originates from the fact that the actual values of the stochastic parameters are not given before the process starts, but instead we will get to know the realizations of the parameters, in our case the outcome of the quality controls, step by step. Every new knowledge after each step can then be used immediately to adjust the schedule.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Reactive scheduling approaches can differ a lot in the implementation since there are no rules, neither which schedule to choose for the initial start of the process nor which algorithm is used to perform the rescheduling. Examples of different variations of the approach for production scheduling problems can be found for instance in Ierapetritou and Li (2009). We will now give a general description of the proposed rescheduling algorithm for the problem at hand.

6.2 Rescheduling algorithm

At first, we need to provide a solution from which we can proceed. This is always determined by a deterministic approach (Ierapetritou and Li (2009)), for instance a worst case solution. By this, we mean a solution of the deterministic model given in chapter 3.2.3 assuming that each job needs the maximum number of tries to be approved by the quality control, in our case 2. Depending on the prior knowledge of the approval probabilities it might also be reasonable not to consider the worst case scenario, but instead assuming rejections only of those jobs for which the rejection probability exceeds a certain limit. However calculated, once a starting solution is given, it provides an initial order in which the jobs enter the system. Thus, the production process can be started, knowing exactly with which jobs the processing begins.

The initial schedule does not only deliver the information, which jobs enter the system first but we can also determine precisely when the first testing takes place and, more importantly, the point in time at which the outcome of this quality control will be given. This is because we assume that all processing times up to this point are deterministic and known in advance.

We can visualize the possible combinations of quality controls' outcomes in form of a scenario tree, such as figure 5.3 presented in chapter 5.2.1. The only difference is that in this approach, the position of the jobs' nodes from left to right is determined by the order in which the quality controls are scheduled. This means that the node the farthest to the left belongs to the job which undergoes the first quality control of the process, i.e. not necessarily job 1 nor the first job that enters the system. Two branches leave each node, which reflects the fact that the result of the quality control is a stochastic variable with only two possible outcomes, rejection or approval.

Once the outcome of the first quality control is known, it is clear which of the two branches can be removed from consideration and thus, half of the scenarios do not even need to be considered anymore. Nevertheless, the moment the outcome of the quality control is known, it is necessary to decide, how to proceed, based on this outcome. Therefore, before this point we need to establish an "optimal" schedule in both cases. We have to provide two

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

optional schedules which are determined according to the chosen rescheduling algorithm. This algorithm takes all allocations and sequences up to the point of the quality control as given and is thus only able to adapt the rest of the schedule. As mentioned above, there are various possibilities for the strategy which is used for rescheduling and it highly depends on the problem as well as the problem structure that is considered. For our problem, we want use the stochastic model introduced in chapter 5. We have argued earlier that the stochastic approach is not able to perform in such a way that it will be useful for practical purposes due to the mere number of scenarios which all have to be considered. Nevertheless, this rescheduling approach provides the opportunity of still making use of the stochastic approach. This can be done by not taking all possible scenarios into account, but instead we will only look ahead a certain number of quality controls, hence ignoring the different scenarios for the quality controls after this point. This reduces drastically the number of scenarios considered for the calculation of a schedule based on the stochastic model.

Depending on the outcome of the quality control either one of the new schedules is chosen. Then, the production proceeds accordingly. The revised production plan is now carried out until the next quality control occurs. This moment again can be determined exactly once it is known which schedule comes into action after the quality control.

Proceeding in the way described above will eventually lead to a production plan that reacts on the outcome of the quality control. It is thus possible to adjust dynamically the schedule in order to either make use of the opening up of machine times due to an unexpected approval of the quality control or to avoid infeasibility and long waiting times by incorporating additional operations which become necessary because of an unexpected rejection.

Example 6.1

In this chapter we will apply the proposed algorithm step by step to the same small scheduling problem we solved with the stochastic approach. Again, we assume that we have 3 jobs that need to be scheduled, the processing times are given in table 4.1 and we do not consider any setup times. The probabilities for rejection or approval are according to the scenario tree in figure 5.3.

Step 1 We need to provide a starting solution in order to decide how to start the production process. We thus solve the worst case scenario with the deterministic model where each job has to complete each task of the operation sequence $[1, 2, 3, 2, 3, 4]$. This corresponds to scenario 8 in the scenario tree depicted below in figure 6.2. The solution can easily be found by passing the vector $\mathsf{scenario} = [2,2,2]$ to the function

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

detModel. The resulting schedule is shown in figure 6.1.

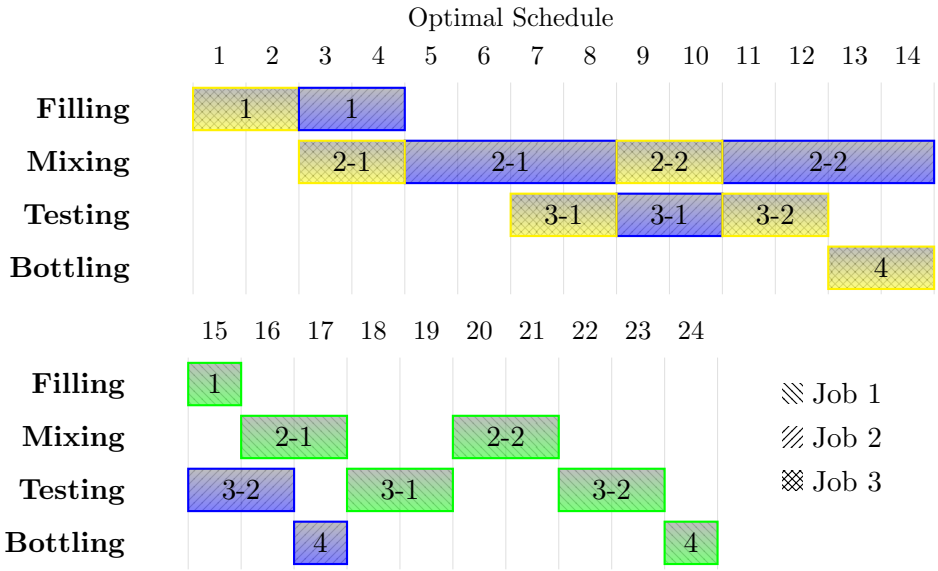

Figure 6.1: The optimal schedule in the worst case scenario of two tries per job, own drawing

As we have mentioned above, considering the probabilities of rejection, it might seem more intuitive to assume that job 2 is approved the first time, since it is rejected only with a very small probability of 0.1. Nevertheless, we chose the pessimistic approach, with the idea of starting with the worst case scenario and improve the schedule in case of unexpected approvals.

Step 2 We can now initiate the production process by starting with the first operation of job 3 on machine 1. After that, job 2 continues on this machine. The schedule also tells that job 3 is the first job to start testing, which will be completed in period 8. The scenario tree with the order of quality controls from left to right according to the schedule is depicted in figure 6.2.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

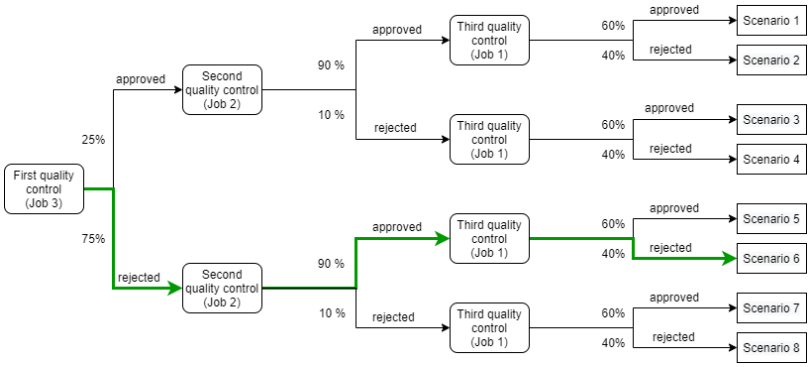

Figure 6.2: Scenario tree for the worst case schedule of **Example 6.1**, own drawing

Step 3 We need to make sure that at the end of period 6 we have two schedules on hand, one if job 3 gets rejected by the quality control, one for the case it is approved. This means that we want to find a schedule with the smallest make span for the upper branch as well as for the lower branch, given the status quo in time period 6, that job 3 has completed the testing and job 2 is being processed by the mixing machine with 2 time periods left until completion. These schedules could again be obtained by using the deterministic model, assuming each missing job needs to go through quality control twice which corresponds to scenarios 4 or 8 depending on the outcome of the first quality control. Note that if we keep using the worst case deterministic approach, the schedule for job 3 being rejected is equivalent to the starting schedule, such that in fact we only need to calculate one additional schedule.

Instead of using the worst case deterministic approach again, we could also resort to the stochastic approach from chapter 5. In this model, the number of scenarios is quite small and since we need an optimal schedule for both branches separately, we only have to calculate the schedule based on 4 scenarios each. We could also consider not taking all possible scenarios into account. This alternative is not reasonable in this small scale example, but would be very useful and necessary when applying the stochastic rescheduling in situations with bigger problem sizes and resulting number of scenarios.

The new schedule was found within the function *stochModel_withStartingSolution* which solves the same stochastic optimization problem as *stochModel*, but initially sets the values for the allocations that cannot be changed anymore because either the operations are already completed or in progress. In order to implement this, we provide the function *stochModel_withStartingSolution* the completion times and allocation matrices. It will also receive the number of quality controls already passed,

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

such that the function is able to calculate the corresponding time period of the current quality control in order to know up to which point in time the original schedule has to be adopted. We will apply the function *stochModel_withStartingSolution* twice to obtain one schedule for the upper branch, one for the other. In the first case, we will pass the corresponding probabilities and processing times for scenario 1-4, in the latter for scenario 5-8 according to the scenario tree in figure 6.2. It is important to note, that the order of the jobs in the scenario tree changed compared to the one in **Example 5.2**. The results of applying the stochastic model to both scenario sets are as follows.

If job 3 is approved, it can be scheduled to be bottled in the next possible period and as soon as this is done, job 1 can enter the system. Job 2 proceeds as before and we will obtain a result of the quality control in period 10. The schedule up to this point is displayed in figure 6.3 where the differences to the previous schedule are marked with a red border.

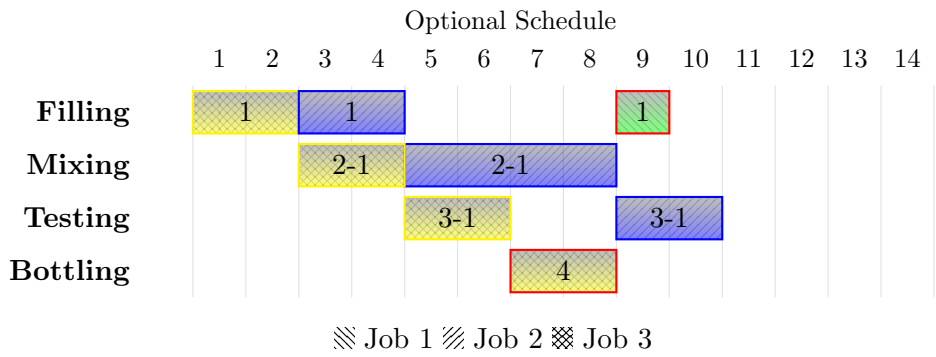

Figure 6.3: The schedule until the next quality control in case job 3 is approved, own drawing

This shows that the schedule can adapt to the new situation in case of an approval of job 3 by the quality control. It is important to note, that operation 2 of job 1, does not start right away in period 10, which would be theoretically possible. This is due to the fact that job 2 is only approved with a probability of 60%, and therefore the allocation of the stochastic approach postpones operation 2 of job 1 to have a buffer for the rework mixing operation of job 2.

In the case, that job 3 is rejected, the schedule until the next quality control is equal to the original schedule till this event in period 10.

In this small example, the order in which the jobs start processing cannot change

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

anymore since at the time the first quality control occurs already two jobs are in the system. However, in problems with a lot more jobs the order in which the not yet processing jobs start might indeed change, although in Cott and Macchietto (1989) it is emphasized that this does not have any effect on the value of the make span. Thus, this will not be useful in problems with the make span as the objective value. It is also important to notice that in this case, the scenario tree does not change either, because the next quality control stays the first quality control of job 2.

Step 4 The moment job 3 is done with testing and the outcome is known, we can either go on until the next quality control occurs with the schedule for rejection, which in this case coincides with the original schedule such as the one in figure 6.1 or we adapt the production plan to continue with the second optional schedule depicted in figure 6.3.

Step 5 We can now tell again, at what time the next quality control will occur and when its outcome will become known. Until then, the two optional schedules have to be calculated and used depending on the outcome. This repetition of Steps 3 and 4 goes on until the last job has passed the quality control and thus, the rest of the production plan is already determined by the current schedule.

We continue the example we started in step 3 and 4, assuming that the first quality control rejects the batch of job 3 and job 2 is approved. The third job to go through the quality control where the outcome is uncertain, which is job 1, is rejected. Note, that the second time job 3 goes through the quality control a positive outcome is certain and thus, there is no need to calculate an optional schedule anymore. In the scenario tree figure 6.2 the actual scenario has already been marked with a green line. In the end, by starting with the worst case scenario schedule and rescheduling with the stochastic approach, we obtain the following actual realization of figure 6.1.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

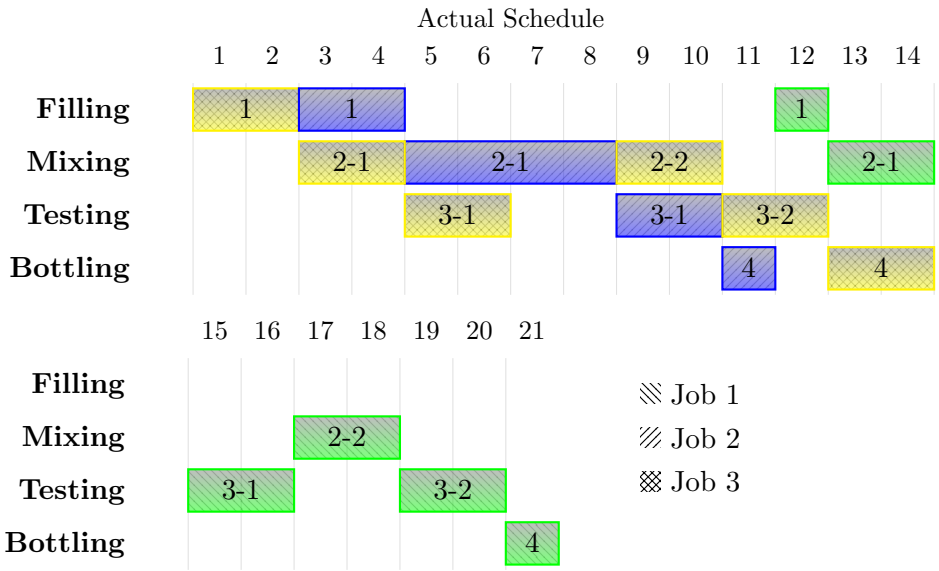

Figure 6.4: The actual realization of the process plan for **Example 6.2**, own drawing

We see that the actual production process took 21 time periods. This production plan has a shorter make span compared to the worst case schedule because job 2 has not been rejected by the quality control. Nevertheless, we were not able to reduce the make span by the total amount of 4 periods that we actually saved by not having to rework job 2. This is due to the fact, that the change in schedule only occurred quite late in the production progress such that no optimizing changes could have been made to benefit from this time savings.

6.3 Extensions

The approach presented above is only a basic idea on how to reduce the number of scenarios that need to be handled at once. There are various extensions and variations of this approach that will generalize as well as enhance the solution. Examples of starting solutions as well as rescheduling algorithms presented in the literature are summarized in Ierapetritou and Li (2009).

As for our problem at hand, the rescheduling approach presented above offers one big opportunity for generalization. In the considerations above we restricted the maximum number of tries for the batches to 2. This restriction can be loosened by dynamically expanding the scenario tree after every quality control. This means that after each rejection, the job is added to the scenario tree once again, such that the next time the job goes through the quality control the outcome is a random variable again.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.4 Problems of this approach

We have already mentioned above, that in problems of realistic size with hundreds of jobs, the rescheduling algorithm might change the order in which the jobs start processing in the middle of the ongoing production. In chapter 2.2.1, we have stated that we do not need to consider the provision of input materials for the scheduling decision. Instead, it is organized after the schedule has been determined. However, due to the possible rescheduling it might occur that materials are needed at a different point in time than originally planned, which becomes a problem especially if the input materials have a limited storability. Thus, it might be necessary to restrict the rescheduling algorithm, taking into account the supply of the input materials.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

7 Conclusion

In this thesis we have addressed various exact methods for finding a solution to a flexible job-shop scheduling problem arising in the chemical industry.

At first, we carefully examined the structure of the production process and its inherent constraints which need to be fulfilled in order to produce a feasible schedule which could actually be implemented in reality. We established the availability of vessels as a major bottleneck of the production process and the presence of uncertainty was discussed.

After the detailed problem description, we developed a deterministic model such that the solution provides a feasible and optimal schedule for fixed operation plans, i.e. for a fixed number of recirculations of each job. The structure of the model was chosen in such a way to include sequence dependent setup times while staying a linear model. In order for the model to incorporate the number of vessels that can be used simultaneously as a constraint, we added a system of constraints with auxiliary variables to keep track of the jobs which are in the processing at the same time. Also, an allocation of each job to its vessel, in which it is processed, is provided by the model.

In the rest of this thesis, three approaches which consider the uncertainty that arises through the stochastic outcome of the quality control were presented. They cover the most important ideas present in the literature to find a solution to a flexible job-shop problem, in which processing times are uncertain. In the first approach, we were focusing on finding a schedule that stays feasible even in the event of perturbations in the data, i.e. batches getting rejected by the quality control, and still providing a solution with objective values as optimal as possible. We applied a solution approach of a flexible job-shop scheduling problem with uncertain processing times provided in the literature. However, since the processing times in our problem can only attain 2 discrete values instead of having of a continuous domain, it retains the applicability of this approach to our problem.

The robust concept was followed by a stochastic approach. With the deterministic model as a basis it considered all scenarios, i.e. all possible combinations of operation plans. The stochastic model then aimed at minimizing the expected value of the make span over all those scenarios. According to its definition, the solution of the stochastic model always provides a robust schedule, in which only the actual time slots of processing vary. In contrast to the robust approach we were able to take into account the additional information

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

of the probabilities with which each scenario occurs.

At last, a reactive scheduling approach was presented. The production process is observed while it is working and the moment an outcome of a quality control is known, corrective actions can be taken in order to repair or enhance the future part of the schedule by adapting it. This approach made use of the deterministic as well as the stochastic model developed in the preceding chapters.

In conclusion, the following results can be established from this thesis. The deterministic approach finds a feasible production plan, that can be implemented in reality since it adheres all conditions and restrictions we assumed in the problem description. However, this production plan lacks in robustness regarding the variability in operation plans due to the uncertain outcomes of the quality control. Thus, we were able to find a robust schedule that finds a solution which deteriorates less for different scenarios compared to the deterministic (worst case) solution. The stochastic approach results in a model robust schedule, which stays feasible for all scenarios, but takes into consideration the probabilities of rejection by the quality control. In the small example that we have provided it was shown that the stochastic approach outperformed solutions of the deterministic as well as robust approach. It will therefore be the first and most reliable choice in theory provided that rejection probabilities are known. However, this and all other approaches pose an impossible computational challenge for problems of real world size due to the immense number of variables and constraints. Thus, in order to effectively use these approaches for practical purposes, it will be inevitable to draw on hybrid or heuristic algorithms to find a solution to the models in a reasonable amount of time. Further research directions might therefore be to find a heuristic solution algorithm adjusted to the model formulation at hand which is able to deal with the large number of binary variables. One might also explore approaches that find a different way of including the uncertainty of the quality control instead of using the processing times.

Apart from these three approaches, we have also presented a reactive schedule that allows intervening in the production plan while the production is already ongoing. Thus, due to the constant update and improvement of the schedule the result cannot be legitimately compared to the ones of the other approaches discussed. If there exists the possibility to implement such a rescheduling procedure, it would be very reasonable to chose this reactive approach. It offers the possibility of including the good results of the stochastic approach but limits the number of scenarios that have to be considered at the same time. However, it will be subject to further research to compare the performance for different starting solutions as well as rescheduling strategies.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek WIEN Your knowledge hub

Bibliography

S. Ahmed. Chapter 29: Stochastic optimization. In *Advances and Trends in Optimization with Engineering Applications*, pages 379–391. SIAM, 2017.

J. Balasubramanian and I. Grossmann. A novel branch and bound algorithm for scheduling flowshop plants with uncertain processing times. *Computers & Chemical Engineering*, 26 (1):41 – 57, 2002. ISSN 0098-1354. doi: https://doi.org/10.1016/S0098-1354(01)00735-9.

D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004. doi: 10.1287/opre.1030.0065.

J. R. Birge and F. Louveaux. *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

A. Caumond, P. Lacomme, A. Moukrim, and N. Tchernev. An milp for scheduling problems in an fms with one vehicle. *European Journal of Operational Research*, 199(3):706 – 722, 2009. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2008.03.051.

B. Cott and S. Macchietto. Minimizing the effects of batch process variability using online schedule modification. *Computers & Chemical Engineering*, 13(1):105 – 113, 1989. ISSN 0098-1354. doi: https://doi.org/10.1016/0098-1354(89)89011-8. Computer Applications to Batch Chemical Processing.

P. Fattahi, M. Saidi Mehrabad, and F. Jolai. Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. *Journal of Intelligent Manufacturing*, 18(3):331–342, Jun 2007. ISSN 1572-8145. doi: 10.1007/s10845-007-0026-8.

C. A. Floudas and X. Lin. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Computers & Chemical Engineering*, 28(11):2109 – 2129, 2004. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2004.05.002.

M. R. Garey, D. S. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, 1(2):117–129, 1976.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

I. E. Grossmann, I. Quesada, R. Raman, and V. T. Voudouris. Mixed-integer optimization techniques for the design and scheduling of batch processes. In G. V. Reklaitis, A. K. Sunol, D. W. T. Rippin, and Ö. Hortaçsu, editors, *Batch Processing Systems Engineering*, pages 451–494, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg. ISBN 978-3-642-60972-5.

S. J. Honkomp, S. Lombardo, O. Rosen, and J. F. Pekny. The curse of reality — why process scheduling optimization problems are difficult in practice. 2000.

IBM. *User Manual IBM Corp.* IBM ILOG CPLEX 12.10, 2019.

M. Ierapetritou and Z. Li. Modeling and managing uncertainty in process planning and scheduling. In *Optimization and logistics challenges in the enterprise*, pages 97–144. Springer, 2009.

A. Iwainsky and W. Wilhelmi. *G*, pages 113–135. Vieweg+Teubner Verlag, Wiesbaden, 1994. ISBN 978-3-322-90716-5. doi: 10.1007/978-3-322-90716-5_7.

V. Jovan. The specifics of production scheduling in process manufacturing. *Elektroniniski vestnik*, 69(5):305–310, 2002.

J. Kallrath. Planning and scheduling in the process industry. *OR Spectrum*, 24(3):219–250, Aug 2002. ISSN 1436-6304. doi: 10.1007/s00291-002-0101-7.

J. Y. Leung. *Handbook of scheduling: algorithms, models, and performance analysis*. CRC Press, 2004.

Z. Li and M. Ierapetritou. Process scheduling under uncertainty: Review and challenges. *Computers & Chemical Engineering*, 32(4-5):715–727, 2008.

X. Lin, S. L. Janak, and C. A. Floudas. A new robust optimization approach for scheduling under uncertainty:: I. bounded uncertainty. *Computers & Chemical Engineering*, 28(6): 1069 – 1085, 2004. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2003.09.020. FOCAPO 2003 Special issue.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in matlab. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

P. B. Luh, Dong Chen, and L. S. Thakur. An effective approach for job-shop scheduling with uncertain processing requirements. *IEEE Transactions on Robotics and Automation*, 15 (2):328–339, April 1999. ISSN 2374-958X. doi: 10.1109/70.760354.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

C. T. Maravelias. General framework and modeling approach classification for chemical production scheduling. *AIChE Journal*, 58(6):1812–1828, 2012. doi: 10.1002/aic.13801.

MATLAB. *9.7.0.1216025 (R2019b)*. The MathWorks Inc., Natick, Massachusetts, 2019.

C. A. Méndez, J. Cerdá, I. E. Grossmann, I. Harjunkoski, and M. Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & Chemical Engineering*, 30(6):913 – 946, 2006. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2006.02.008.

M. Mousakhani. Sequence-dependent setup time flexible job shop scheduling problem to minimise total tardiness. *International Journal of Production Research*, 51(12):3476–3487, 2013.

J. Pinto and I. Grossmann. Assignment and sequencing models of the scheduling of process systems. *Annals of Operations Research - Annals OR*, 81:433–466, 06 1998. doi: 10.1023/A:1018929829086.

M. Šeda. Mathematical models of flow shop and job shop scheduling problems. *World Academy of Science, Engineering and Technology*, 1(31):122–127, 2007.

M. Shafia, M. Aghaee, and A. Jamili. A new mathematical model for the job shop scheduling problem with uncertain processing times. *International Journal of Industrial Engineering Computations*, 2, 04 2011. doi: 10.5267/j.ijiec.2010.03.005.

X.-D. Zhang and H.-S. Yan. Integrated optimization of production planning and scheduling for a kind of job-shop. *The International Journal of Advanced Manufacturing Technology*, 26(7-8):876–886, 2005.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Appendices

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Listing 7.1: MATLAB function implementing the deterministic model

```
function [makespan, schedule, allocation, vesselallocation] =
    detModel(n, scenario, v_max, m1, m2, m4, alpha, p, set)

m = m1 + m2 + v_max + m4; % total number of machines

O = cell(1,3);
for i = 1:n-1
    temp = repmat([2,3],1,scenario(i));
    % operation plans according to number of recirculations
    O{i} = [1,temp,4];
end

l = zeros(1,n);
l(1,1) = 1;
for i = 1:length(O)
    l(i+1) = length(O{i}); % number of operations per job
end
lj_max = max(l(2:n)); % maximal number of operations

L = 10000; % large number

% variables
x = binvar(n, lj_max, n, lj_max, m, 'full');
a = binvar(n-1, n-1, 'full');
b = binvar(n-1, n-1, 'full');
w = binvar(n-1, n-1, 'full');
d = binvar(n-1, v_max, 'full');
C = sdpvar(n, lj_max, 'full');
C_max = sdpvar(1);

% non-negativity constraints
Constraints = [C(:) >= 0];

% constraints for objective value
for j = 2:n
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
Constraints = [Constraints, C_max >= C(j,l(j))];
end

% allocation constraints
for j = 1:n
    for k = (l(j)+1):lj_max
        Constraints = [Constraints,
            x(j,k,:,:,:) == 0,
            x(:,:,j,k,:) == 0];
    end
end

for j = 2:n
    for k = 1:l(j)
         Constraints = [Constraints,
(sum(sum(sum(x(j,k,:,:,:),5),4),3) == 1),
sum(sum(x(j,k,:,:,1:m),4),3) <= alpha{j-1}(O{j-1}(k),1:m),
(sum(sum(sum(x(2:n,:,j,:,:),5),2),1) <= 1),
sum(sum(x(2:n,:,j,k,1:m),2),1) <= sum(sum(x(j,k,:,:,1:m),4),3)
   ];
    end
end

Constraints = [Constraints,
sum(sum(x(2:n ,: ,1 ,1 ,1:m) ,2) ,1) <= 1];



% sequence constraints
for j = 2:n
    Constraints = [Constraints, C(j,1) >= p(j,1)];
    for k = 2:l(j)
        Constraints = [Constraints,
        C(j,k) >= C(j,(k-1)) + p(j,O{j-1}(k))];
    end
end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
for j = 2:n
    for k = 1:l(j)
        for h = 1:n
            for z = 1:l(h)
                Constraints = [Constraints,
C(j,k) >= C(h,z) + sum(x(j,k,h,z,:).*set(j,k,h,z,:),5)
    + p(j,O{j-1}(k)) - L*(1-sum(x(j,k,h,z,:),5))];
            end
        end
    end
end

% vessel constraints
for j = 2:n
    for h = 2:n
        if j ~= h
    Constraints = [Constraints,
    (C(h,1)-p(h,1)) - (C(j,1)-p(j,1)) + L*(1-a(j-1,h-1)) >= 0,
    (C(j,1)-p(j,1)) - (C(h,1)-p(h,1)) + L *(a(j-1,h-1)) >= 0,
    (C(j,1)-p(j,1)) - C(h,l(h)) + L*(1-b(j-1,h-1)) >= 0,
    C(h,l(h)) - (C(j,1)-p(j,1)) + L*(b(j-1,h-1)) >= 0,
    1 >= a(j-1,h-1)+a(h-1,j-1)
    w(j-1,h-1) - a(j-1,h-1) - b(j-1,h-1) <= 0,
    a(j-1,h-1) - w(j-1,h-1) <= 0,
    b(j-1,h-1) - w(j-1,h-1) <= 0,
    d(j-1,1:v_max) + d(h-1,1:v_max) <= w(j-1,h-1)+w(h-1,j-1)];
        end
    end
end


for j = 1:(n-1)
    Constraints = [Constraints,
    sum(w(j,1:(n-1)~=(j)),2) >= (n-1) - v_max];
end
```


Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

```
for j = 1:(n-1)
    Constraints = [Constraints, sum(d(j,:),2) == 1];
end

% objective function
Objective = C_max;

% set solver
options = sdpsettings('solver','cplex');

% solve the problem
sol = optimize(Constraints, Objective, options);

% return values
makespan = value(C_max);
schedule = value(C);
allocation = value(x);
vesselallocation = value(d);
end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Listing 7.2: MATLAB function implementing the deterministic model for a given allocation

```
function [makespan, schedule, allocation] =
    detModel_withStartingSolution(n, scenario, v_max, m1,m2,m4,
    alpha,p,set,C_alt,x_alt,number)

m = m1 + m2 + v_max + m4; % total number of machines

O = cell(1,3);
for i = 1:n-1
    temp = repmat([2,3],1,scenario(i));
    % operation plans according to number of recirculations
    O{i} = [1,temp,4];
end

l = zeros(1,n);
l(1,1) = 1;
for i = 1:length(O)
    l(i+1) = length(O{i}); % number of operations per job
end
lj_max = max(l(2:n)); % maximal number of operations

L = 10000; % large number

% variables
% x = binvar(n, lj_max, n, lj_max, m, 'full');
a = binvar(n-1, n-1, 'full');
b = binvar(n-1, n-1, 'full');
w = binvar(n-1, n-1, 'full');
d = binvar(n-1, v_max, 'full');
C = sdpvar(n, lj_max, 'full');
C_max = sdpvar(1);

% non-negativity constraints
 Constraints = [C(:) >= 0];


```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

```
% constraints for objective value
for j = 2:n
Constraints = [Constraints, C_max >= C(j,l(j))];
end

% sequence constraints
for j = 2:n
    Constraints = [Constraints, C(j,1) >= p(j,1)];
    for k = 2:l(j)
        Constraints = [Constraints,
            C(j,k) >= C(j,(k-1)) + p(j,k)];
    end
end

for j = 2:n
    for k = 1:l(j)
        for h = 1:n
            for z = 1:l(h)
                Constraints = [Constraints,
C(j,k) >= C(h,z) + p(j,k)- L*(1-sum(x_alt(j,k,h,z,:),5))];
            end
        end
    end
end

% vessel auxilliary constraints
for j = 2:n
    for h = 2:n
        if j ~= h
    Constraints = [Constraints,
    (C(h,1)-p(h,1)) - (C(j,1)-p(j,1)) + L*(1-a(j-1,h-1)) >= 0,
    (C(j,1)-p(j,1)) - (C(h,1)-p(h,1)) + L *(a(j-1,h-1)) >= 0,
    (C(j,1)-p(j,1)) - C(h,l(h)) + L*(1-b(j-1,h-1)) >= 0,
    C(h,l(h)) - (C(j,1)-p(j,1)) + L*(b(j-1,h-1)) >= 0,
    1 >= a(j-1,h-1)+a(h-1,j-1)];
        end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
    end
end

% vessel constraints
for j = 1:(n-1)
    for h = 1:(n-1)
        if j~=h
            Constraints = [Constraints,
            w(j,h) - a(j,h) - b(j,h) <= 0,
            a(j,h) - w(j,h) <= 0,
            b(j,h) - w(j,h) <= 0];
        end
    end
end

for j = 1:(n-1)
    Constraints = [Constraints,
        sum(w(j,1:(n-1)~=(j)),2) >= (n-1) - v_max];
end

% vessel allocation
for j = 1:(n-1)
    for h = 1:(n-1)
        if j~=h
            Constraints = [Constraints,
                d(j,1:v_max) + d(h,1:v_max) <= w(j,h)+w(h,j)];
        end
    end
end

for j = 1:(n-1)
    Constraints = [Constraints, sum(d(j,:),2) == 1];
end

% objective function
Objective = C_max;
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
% set solver
options = sdpsettings('solver','cplex');

% solve the problem
sol = optimize(Constraints, Objective, options);

% return values
makespan = value(C_max);
schedule = value(C);
allocation = value(x_alt);
vesselallocation = value(d);
end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Listing 7.3: MATLAB function implementing the robust model

```
function [makespan, schedule, allocation, vesselallocation] =
    detModel_robust(n, scenario, v_max, m1, m2, m4, alpha, p,
    Gamma)

m = m1 + m2 + v_max + m4; % total number of machines

O = cell(1,3);
for i = 1:n-1
    temp = repmat([2,3],1,scenario(i));
    % operation plans according to number of recirculations
    O{i} = [1,temp,6];
end

l = zeros(1,n);
l(1,1) = 1;
for i = 1:length(O)
    l(i+1) = length(O{i}); % number of operations per job
end
lj_max = max(l(2:n)); % maximal number of operations

L = 10000; % large number

% variables
x = binvar(n, lj_max, n, lj_max, m, 'full');
a = binvar(n-1, n-1, 'full');
b = binvar(n-1, n-1, 'full');
w = binvar(n-1, n-1, 'full');
phi = binvar(n, lj_max, n, lj_max, 'full');
rho = binvar(n-1, lj_max, n, lj_max, n, lj_max, m, 'full');
d = binvar(n-1, v_max, 'full');
e = sdpvar(n, lj_max, n, lj_max, m, 'full');
eta = sdpvar(n, lj_max, n, lj_max, m, n-1, 'full');
C = sdpvar(n, lj_max, 'full');
C_max = sdpvar(1);

```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

```
% non-negativity constraints
Constraints = [C(:) >= 0, C(1,1) == 0, eta(:) >= 0, e(:) >= 0];


% constraints for objective value
for j = 1:n
Constraints = [Constraints, C_max >= C(j,l(j))];
end

% allocation constraints
for j = 1:n
    for k = 1:lj_max
        for h = 1:n
            for z = 1:lj_max
                Constraints = [Constraints,
                x(j,k,h,z,1:m) + x(h,z,j,k,1:m) <= 1];
            end
        end
    end
end


for j = 1:n
    for k = (l(j)+1):lj_max
        Constraints = [Constraints, x(j,k,:,:,:) == 0];
    end
end

for h = 1:n
    for z = (l(h)+1):lj_max
        Constraints = [Constraints, x(:,:,h,z,:) == 0];
    end
end

for j = 1:n
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
    for k = 1:lj_max
Constraints = [Constraints, x(j,k,j,k,:) == 0];
    end
end

for j = 2:n
    for k = 1:l(j)
        Constraints = [Constraints,
        (sum(sum(sum(x(j,k,:,:,:),5),4),3) == 1)];
    end
end

for j = 2:n
    for k = 1:l(j)
    Constraints = [Constraints,
    sum(sum(x(j,k,:,:,1:m),4),3) <= alpha{j-1}(k,1:m)];
    end
end

for h = 2:n
    for z = 1:l(h)
     Constraints = [Constraints,
    (sum(sum(sum(x(2:n,:,h,z,:),5),2),1) <= 1)];
    end
end

Constraints = [Constraints,
    sum(sum(x(2:n,:,1,1,1:m),2),1) <= 1];


for h = 2:n
    for z = 1:l(h)
Constraints = [Constraints,
sum(sum(x(2:n,:,h,z,1:m),2),1) <= sum(sum(x(h,z,:,:,1:m),4),3)
   ];
    end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
end

% sequence constraints
for j = 2:n
    Constraints = [Constraints, C(j,1) >= p(1,1,1,1,j,1,1)];
    for k = 2:l(j)
        Constraints = [Constraints,
        C(j,k) >= C(j,(k-1)) + p(1,1,1,1,j,O{j-1}(k),1)];
    end
end

for j = 1:n
    for k = 1:l(j)
        for h = 1:n
            for z = 1:l(h)
                Constraints = [Constraints,
                C(h,z) - C(j,k) + L*phi(j,k,h,z) >= 0,
                C(j,k) - C(h,z) + L*(1-phi(j,k,h,z)) >= 0,
                phi(j,k,h,z) + phi(h,z,j,k) >= 1];
            end
        end
    end
end

for j = 2:n
    for k = 1:l(j)
        for h = 1:n
            for z = 1:l(h)
                for f = 1:n
                    for g = 1:l(f)
                        for i = 1:m
Constraints = [Constraints,
L*rho(j-1,k,h,z,f,g,i) >= sum(sum(x(f,g,:,:,i),4),3)
    + phi(f,g,h,z) + phi(j,k,f,g)- 2,
-L*(1-rho(j-1,k,h,z,f,g,i)) <= sum(sum(x(f,g,:,:,i),4),3)
    + phi(f,g,h,z) + phi(j,k,f,g) - 3];
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
if (h > 1)
    Constraints = [Constraints,
    x(j,k,h,z,i) <= 1-rho(h-1,z,1,1,j,k,i)];
                                end
                            end
                        end
                    end
                end
            end
        end
end

for j = 2:n
    for k = 1:l(j)
        for h = 1:n
            for z = 1:l(h)
                for i = 1:m
                    sum1 = 0;
                    for j1 = 2:n
                        for l1 = 1:l(j1)
% sum of processing times for operations not in P
if (O{j1-1}(l1) ~= 2 && O{j1-1}(l1) ~= 3
    && (j1 ~= h || l1 ~= z))
sum1 = sum1 + rho(j-1,k,h,z,j1,l1,i)*p(j,k,h,z,j1,l1,i);
                                end
                            end
                        end
Constraints = [Constraints,
C(j,k) >= C(h,z) + sum1 + sum(eta(j,k,h,z,i,:),6)
    + Gamma*e(j,k,h,z,i) - L*(1-rho(j-1,k,1,1,h,z,i))
    - L*(1-sum(sum(x(j,k,:,:,i),4),3))];
                            for j2 = 2:n
                                    sum2= 0;
                                for l2 = 1:l(j2)
% sum of processing times for operations in P
if (O{j2-1}(l2) ~= 1 && O{j2-1}(l2) ~= 6
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
    && (j2 ~= h || l2 ~= z))
sum2 = sum2 + rho(j-1,k,h,z,j2,l2,i).*p(j,k,h,z,j2,O{j2-1}(l2)
   ,i);
                                    end
                                end
Constraints = [Constraints,
e(j,k,h,z,i) + eta(j,k,h,z,i,j2-1) >= sum2];
                            end
                    end
                end
            end
        end
end

% vessel auxilliary constraints
for j = 2:n
    for h = 2:n
        if j ~= h
Constraints = [Constraints,
(C(h,1)-p(1,1,1,1,h,1,1)) - (C(j,1)-p(1,1,1,1,j,1,1))
    + L*(1-a(j-1,h-1)) >= 0,
(C(j,1)-p(1,1,1,1,j,1,1)) - (C(h,1)-p(1,1,1,1,h,1,1))
    + L *(a(j-1,h-1)) >= 0,
(C(j,1)-p(1,1,1,1,j,1,1)) - C(h,l(h)) + L*(1-b(j-1,h-1)) >= 0,
C(h,l(h)) - (C(j,1)-p(1,1,1,1,j,1,1)) + L*(b(j-1,h-1)) >= 0,
1 >= a(j-1,h-1)+a(h-1,j-1)];
        end
    end
end

% vessel constraints
for j = 1:(n-1)
    for h = 1:(n-1)
        if j~=h
            Constraints = [Constraints,
            w(j,h) - a(j,h) - b(j,h) <= 0,
```


Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

```
            a(j,h) - w(j,h) <= 0,
            b(j,h) - w(j,h) <= 0];
        end
    end
end

for j = 1:(n-1)
    Constraints = [Constraints,
        sum(w(j,1:(n-1)~=(j)),2) >= (n-1) - v_max];
end

% vessel allocation
for j = 1:(n-1)
    for h = 1:(n-1)
        if j~=h
            Constraints = [Constraints,
                d(j,1:v_max) + d(h,1:v_max) <= w(j,h)+w(h,j)];
        end
    end
end

for j = 1:(n-1)
    Constraints = [Constraints, sum(d(j,:),2) == 1];
end

% objective function
Objective = C_max;

% set solver
options = sdpsettings('solver','cplex');

% solve the problem
sol = optimize(Constraints, Objective, options);

% return values
makespan = value(C_max);
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

```
schedule = value(C);
allocation = value(x);
vesselallocation = value(rho);
end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Listing 7.4: MATLAB function implementing the stochastic model

```
function[makespan, schedule, allocation, vesselallocation] =
    stochModel(n, v_max, m1, m2, m4, alpha, p, setup, prob,
    C_alt, x_alt, number)

m = m1 + m2 + v_max + m4; % total number of machines

for i = 1:n-1
    temp = repmat([2,3],1,2);
    O{i} = [1,temp,4]; % operation plans
end

l = zeros(1,n);
l(1,1) = 1;
for i = 1:length(O)
    l(i+1) = length(O{i}); % number of operations per job
end

lj_max = max(l(2:n)); % maximal number of operations

s_max = length(prob); % number of scenarios

L = 10000; % large number

% variables
x = binvar(n, lj_max, n, lj_max, m,'full');
a = binvar(n-1, n-1, s_max, 'full');
b = binvar(n-1, n-1, s_max, 'full');
w = binvar(n-1, n-1, s_max, 'full');
d = binvar(n-1, v_max, 'full');

C = sdpvar(n, lj_max,s_max,'full');
C_max = sdpvar(s_max,1);

% non-negativity constraints
Constraints = [C(:,:) >= 0];
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
% allocation constraints independent of scenario
for j = 1:n
    for k = (l(j)+1):lj_max
        Constraints = [Constraints, x(j,k,:,:,:) == 0];
    end
end

for h = 1:n
    for z = (l(h)+1):lj_max
        Constraints = [Constraints, x(:,:,h,z,:) == 0];
    end
end

for j = 2:n
    for k = 1:l(j)
Constraints = [Constraints,
(sum(sum(sum(x(j,k,:,:,:),5),4),3) == 1)];
    end
end

for j = 2:n
    for k = 1:l(j)
        for i = 1:m
Constraints = [Constraints,
sum(sum(x(j,k,:,:,i),4),3)  <= alpha{j-1}(O{j-1}(k),i)];
        end
    end
end

for h = 2:n
Constraints = [Constraints,
(sum(sum(sum(x(2:n,:,h,1:l(h),:),5),2),1) <= 1)];
end

for i = 1:m
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
  Constraints = [Constraints,
  sum(sum(x(2:n,:,1,1,i),2),1) <= 1];
end

for h = 2:n
    for z = 1:l(h)
        for i = 1:m
Constraints = [Constraints,
sum(sum(x(2:n,:,h,z,i),2),1) <= sum(sum(x(h,z,:,:,i),4),3)];
        end
    end
end

% constraints containing values that depend on the scenario
   have to  be
% added for each scenario
for s = 1:s_max

for j = 2:n
    Constraints = [Constraints, C_max(s) >= C(j,l(j),s)];
end

% sequence constraints
for j = 2:n
    Constraints = [Constraints, C(j,1,s) >= p(j,1,s)];
    for k = 2:l(j)
        Constraints = [Constraints,
        C(j,k,s) >= C(j,(k-1),s) + p(j,k,s)];
    end
end

for j = 2:n
    for k = 1:l(j)
        for h = 1:n
            for z = 1:l(h)
Constraints = [Constraints,
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
C(j,k,s) >= C(h,z,s)+ sum(x(j,k,h,z,:).*setup(j,k,h,z,:),5)
    + p(j,k,s) - L*(1-sum(x(j,k,h,z,:),5))];
            end
        end
    end
end

% vessel constraints
for j = 2:n
    for h = 2:n
        if j ~= h
Constraints = [Constraints,
(C(h,1,s)-p(h,1,s)) - (C(j,1,s)-p(j,1,s)) + L*(1-a(j-1,h-1,s))
    >= 0,
(C(j,1,s)-p(j,1,s)) - (C(h,1,s)-p(h,1,s)) + L*(a(j-1,h-1,s))
   >= 0,
(C(j,1,s)-p(j,1,s)) - C(h,l(h),s) + L*(1-b(j-1,h-1,s)) >= 0,
C(h,l(h),s) - (C(j,1,s)-p(j,1,s)) + L*(b(j-1,h-1,s)) >= 0,
1 >= a(j-1,h-1,s)+a(h-1,j-1,s),
w(j-1,h-1,s) - a(j-1,h-1,s) - b(j-1,h-1,s) <= 0,
a(j-1,h-1,s) - w(j-1,h-1,s) <= 0,
b(j-1,h-1,s) - w(j-1,h-1,s) <= 0,
d(j-1,1:v_max) + d(h-1,1:v_max) <= w(j-1,h-1,s)+w(h-1,j-1,s)];
        end
    end
end

for j = 1:(n-1)
    Constraints = [Constraints,
    sum(w(j,1:(n-1)~=(j),s),2) >= (n-1) - v_max];
end

for j = 1:(n-1)
    Constraints = [Constraints, sum(d(j,:),2) == 1];
end

```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
end

% define an objective
Objective = prob * C_max;

% set some options for YALMIP and solver
options = sdpsettings('solver','cplex');

% solve the problem
sol = optimize(Constraints, Objective, options);

% return values
makespan = value(Objective);
schedule = value(C);
allocation = value(x);
vesselallocation = value(d);
end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Listing 7.5: MATLAB function implementing the stochastic model with a given allocation up to a certain number of quality controls

```
function[makespan, schedule, allocation, vesselallocation] =
    stochModel_withStartingSolution(n, v_max, m1, m2, m4, alpha
    , p, setup, prob, C_alt, x_alt, TestingNumber)

m = m1 + m2 + v_max + m4; % total number of machines

for i = 1:n-1
    temp = repmat([2,3],1,2);
    O{i} = [1,temp,4]; % operation plans
end

l = zeros(1,n);
l(1,1) = 1;
for i = 1:length(O)
    l(i+1) = length(O{i}); % number of operations per job
end

lj_max = max(length(O)); % maximal number of operations

s_max = length(prob); % number of scenarios

L = 10000; % large number

% variables
x = binvar(n, lj_max, n, lj_max, m,'full');
a = binvar(n-1, n-1, s_max, 'full');
b = binvar(n-1, n-1, s_max, 'full');
w = binvar(n-1, n-1, s_max, 'full');
d = binvar(n-1, v_max, 'full');

C = sdpvar(n, lj_max,s_max,'full');
C_max = sdpvar(s_max,1);

Constraints = [C(:,:) >= 0];
```


Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

```
33
34 for s = 1:s_max
35 % pre-allocate values to variables in order to represent the
      schedule so
36 % far
37 cuttingperiods = sort(C_alt(2:n,3));
38 cuttingperiod = cuttingperiods(TestingNumber);
39 x_alt(isnan(x_alt))=0;
40
41 for j = 2:n
42     for i = 1:lj_max
43 if (C_alt(j,i) <= cuttingperiod)||
44 (i > 1 && C_alt(j,i-1) < cuttingperiod && C_alt(j,i) >=
      cuttingperiod)
45 Constraints = [Constraints,
46     x(j,i,:,:,1:m) == x_alt(j,i,:,:,1:m)];
47         end
48     end
49 end
50
51 for j = 2:n
52     Constraints = [Constraints, C_max(s) >= C(j,l(j),s)];
53 end
54
55
56 for j = 1:n
57     for k = (l(j)+1):lj_max
58         Constraints = [Constraints, x(j,k,:,:,:) == 0];
59     end
60 end
61 for h = 1:n
62     for z = (l(h)+1):lj_max
63         Constraints = [Constraints, x(:,:,h,z,:) == 0];
64     end
65 end
66
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
for j = 2:n
    for k = 1:l(j)
        Constraints = [Constraints,
        (sum(sum(sum(x(j,k,:,:,:),5),4),3) == 1)];
    end
end

for j = 2:n
    for k = 1:l(j)
        for i = 1:m
        Constraints = [Constraints,
        sum(sum(x(j,k,:,:,i),4),3) <= alpha{j-1}(O{j-1}(k),i)
           ];
        end
    end
end

for h = 2:n
    Constraints = [Constraints,
    (sum(sum(sum(x(2:n,:,h,1:l(h),:),5),2),1) <= 1)];
end

for i = 1:m
    Constraints = [Constraints,
    sum(sum(x(2:n,:,1,1,i),2),1) <= 1];
end

for h = 2:n
    for z = 1:l(h)
        for i = 1:m
            Constraints = [Constraints,
sum(sum(x(2:n,:,h,z,i),2),1) <= sum(sum(x(h,z,:,:,i),4),3)];
        end
    end
end

```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
% sequence constraints
for j = 2:n
    Constraints = [Constraints, C(j,1,s) >= p(j,1,s)];
    for k = 2:l(j)
        Constraints = [Constraints,
        C(j,k,s) >= C(j,(k-1),s) + p(j,k,s)];
    end
end

for j = 2:n
    for k = 1:l(j)
        for h = 1:n
            for z = 1:l(h)
                summe = 1-sum(x(j,k,h,z,:),5);
Constraints = [Constraints,
C(j,k,s) >= C(h,z,s)+ sum(x(j,k,h,z,:).*setup(j,k,h,z,:),5)
    + p(j,k,s) - L*(summe)];
            end
        end
    end
end

% vessel auxilliary constraints

for j = 2:n
    for h = 2:n
        if j ~= h
Constraints = [Constraints,
(C(h,1,s)-p(h,1,s)) - (C(j,1,s)-p(j,1,s)) + L*(1-a(j-1,h-1,s))
    >= 0,
(C(j,1,s)-p(j,1,s)) - (C(h,1,s)-p(h,1,s)) + L*(a(j-1,h-1,s))
   >= 0,
(C(j,1,s)-p(j,1,s)) - C(h,l(h),s) + L*(1-b(j-1,h-1,s)) >= 0,
C(h,l(h),s) - (C(j,1,s)-p(j,1,s)) + L*(b(j-1,h-1,s)) >= 0,
1 >= a(j-1,h-1,s)+a(h-1,j-1,s)];
        end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

```
    end
end

% vessel constraints
for j = 1:(n-1)
    for h = 1:(n-1)
        Constraints = [Constraints,
        w(j,h,s) - a(j,h,s) - b(j,h,s) <= 0,
        a(j,h,s) - w(j,h,s) <= 0,
        b(j,h,s) - w(j,h,s) <= 0];
    end
end

for j = 1:(n-1)
    Constraints = [Constraints,
    sum(w(j,1:(n-1)~=(j),s),2) >= (n-1) - v_max];
end

% vessel allocation
for j = 1:(n-1)
    for h = 1:(n-1)
        Constraints = [Constraints,
        d(j,1:v_max) + d(h,1:v_max) <= w(j,h,s)+w(h,j,s)];
    end
end

for j = 1:(n-1)
    Constraints = [Constraints, sum(d(j,:),2) == 1];
end

end

% define an objective
Objective = prob *C_max;

% set some options for YALMIP and solver
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.

```
options = sdpsettings('solver','cplex');

% solve the problem
sol = optimize(Constraints,Objective,options);

% return values
makespan = value(C_max);
schedule = value(C);
allocation = value(x);
vesselallocation = value(d);
end
```

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar.
The approved original version of this thesis is available in print at TU Wien Bibliothek.