


MSC THESIS



Strengthening a Mixed-Integer Program for Scheduling an Industrial Formulation Plant

Using Facet-Defining Inequalities and a Separation Algorithm

Sanne Oude Veldhuis

Graduation committee:

dr. M. Walter

prof. dr. M.J. Uetz

dr. rer. nat. S. Langer

Role:

Daily Supervisor

Graduation Supervisor

External Committee Member

April 4, 2024

Department of Applied Mathematics

Faculty of Electrical Engineering, Mathematics and Computer Science

Discrete Mathematics and Mathematical Programming (DMMP)

UNIVERSITY OF TWENTE.

Preface

With this thesis I conclude my time studying at the University of Twente. A time in which I finished my bachelor and master in Applied Mathematics and had a great board year at W.S.G. Abacus. I have enjoyed my time here and made some great friends along the way.

I would like to thank Matthias Walter for his supervision and support with this thesis. I have enjoyed our sessions in which we filled many blackboards with equations and had nice discussions about the subject. I really appreciate how involved you were and how quickly you responded to all my questions or draft versions. I also want to thank the other members of my graduation committee for taking the time to read and assess my work.

Last, but not least, I want to thank my family and friends for being there and listening to me talking or complaining about my work or even proofreading my thesis. A special thanks to the people who studied with me or adopted me in their workplace so I could have a space to work. But apart from that, I mainly want to thank my friends for all the fun we had so I did not have to think about my thesis all the time.

Abstract

The problem of solving a scheduling problem for a two-level industrial formulation plants is discussed. An improved version of the mixed-integer program (MIP) introduced by Yfantis (Computer Aided Chemical Engineering, Vol. 46, (2019)) is given. However, the integrality gap in existing MIPs is too large. Hence, the goal is to strengthen the MIP formulation to reduce the integrality gap. The entire model is too large and intricate to immediately improve. So a subproblem is set up that only includes the filling part of the formulation plant. The linking of the indicator variable for the makespan plays a big role in the integrality gap, that is why the focus is on the filling part that includes this indicator variable. For this subproblem, facet-defining inequalities are found, as well as a separation algorithm to implement these specific inequalities. The found inequalities strengthen the MIP formulation.

Keywords: Mixed-integer program, industrial formulation plant, facet, cutting plane

Contents

1	Introduction	3
2	Problem description	6
2.1	Constraints and assumptions	6
2.1.1	Definition of sets, parameters and variables	7
2.2	Initial model	7
2.2.1	Incorrectness of the model	8
2.3	New proposed model	9
2.3.1	Explanation of variables	9
2.3.2	Explanation of the constraints	10
2.4	Comparison of both models	14
3	Model strengthening	15
3.1	Facet-defining inequalities	15
3.1.1	Setting up a subproblem	15
3.1.2	Finding facet-defining inequalities	16
3.2	LP bounds	20
3.2.1	Implementation of the separation algorithm	21
4	Extended strengthening	23
4.1	Disjunctive programming	23
4.2	Extended facet-defining inequality	24
4.3	Separation algorithm	25
5	Conclusion and discussion	27
	Bibliography	29
	Appendices	A-1
A.1	Yfantis' original model	A-1
A.2	Yfantis' fixed model	A-3
A.3	New proposed model	A-5
A.4	Model comparison	A-7

Chapter 1

Introduction

The problem of developing methods for scheduling an industrial formulation plant and optimizing production planning has existed for over 50 years (Kopanos et al., 2010) and is still of value in the production industry. For manufacturers, being able to generate and implement optimal schedules can save a lot of time and money. Unfortunately, there is not one model that can accurately describe all plants. This is because every plant is unique, either in their setup and constraints or even in their objective. The setup and constraints are for example precedence constraints on products, setup times, order-dependent changeover times and restrictions for specific products on machines. The different objectives can for example be to minimize the total cost, the lateness, the changeover and setup times, the total completion time or a weighted combination of these objectives.

This introduces a problem that occurs when optimizing an industrial formulation plant. The more complicated or precise its setup, constraints or objective are, the more difficult it is to accurately model this and the more time consuming it is to solve. In most cases, this implies that a mathematical optimization model can only be implemented on small test instances and not on instances of applicable size. That is why it is interesting to look into strengthening mathematical optimization models.

Every mathematical optimization model is either a discrete-time model or a continuous-time model. There are advantages and disadvantages for both types (Floudas & Lin, 2004). A discrete-time model uses a discretized time horizon, so it is split up into periods of equal length and all scheduling is done at those points in time. The advantage of this is that it is proven to be a reliable and straightforward way of modelling a situation, by choosing if something starts or ends processing at a certain time period. The disadvantage, however, is that the periods have to be sufficiently small to get an accurate representation, which implies that the size of a model can blow up quickly when expanding the time horizon. A continuous-time model on the other hand only includes the times at which an order is planned to start or stop processing, hence creating a smaller model. However, in most instances, it is a lot more difficult to describe a real-life instance with a continuous time model, since the timing of events is variable, which results in difficult modelling structures. That is why we use a discrete-time model over a continuous-time model in this thesis.

Specifically, a *Mixed-Integer Program* (MIP) is made to accurately model the process. A MIP uses binary or integer variables and can also include some continuous variables, opposed to a *Linear Program* (LP) which uses only continuous variables. Not only is there a difference between a MIP and an LP in variable type, but there is also big difference in how well both models can be solved by existing software and algorithms. There are multiple algorithms that are able to solve LPs within reasonable time, since it is proven to be in P (Khachiyan, 1980). However, integer programming is NP-hard (Conforti et al., 2014b), so one cannot expect to find a polynomial time algorithm to solve a MIP.

In this thesis we focus on producing chemicals in an industrial formulation plant. This process consists of mixing raw materials, processing them and then packaging them. For the mathematical model, it does not matter what the orders or products look like, only the properties of the process are important. This implies that our research can be translated to other industries with similar plant setups, such as the food, beverage or pharmaceuticals industry.

The industrial formulation plant studied in this research contains two main phases. First, there are formulation lines, which process the raw material into unpackaged product. Next, the unpackaged product is filled into packaging by filling stations. If needed, unpackaged product can be intermediately stored in buffer tanks. These three parts are connected by a transfer panel. A schematic depiction of the plant is given in Figure 1.1.

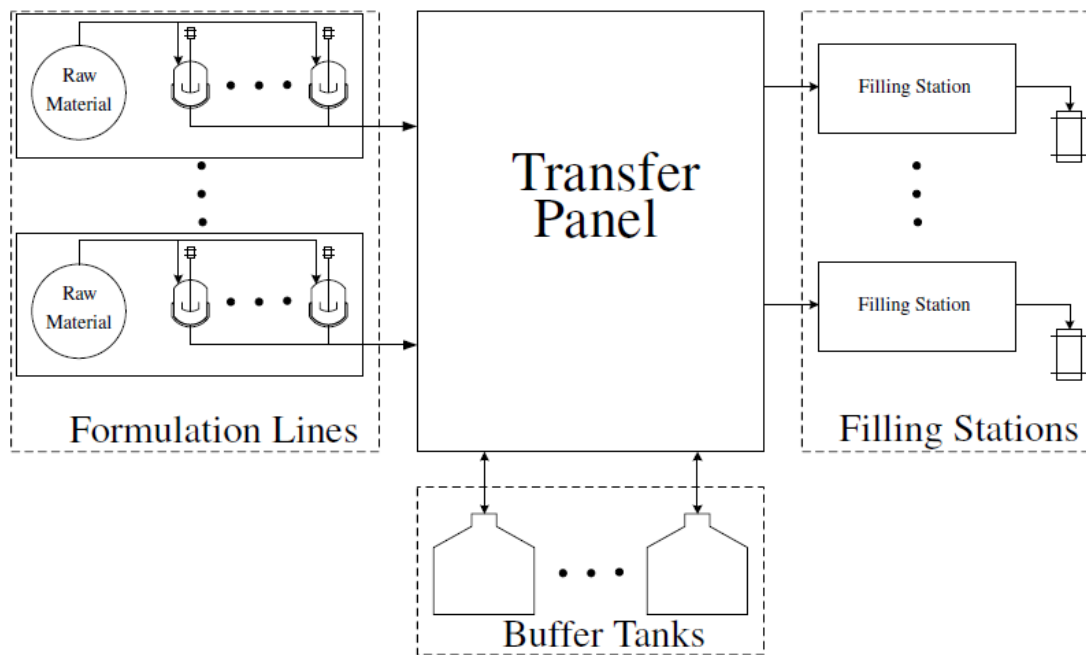


Figure 1.1: A schematic depiction of the industrial formulation plant. (Yfantis et al., 2019)

In this plant, there are sequence-dependent changeover times on both the formulation lines and filling stations. The processing times differ per product and machine. There are no precedence constraints and preemption is possible on the formulation lines.

As mentioned, a lot of time and money can be saved by optimizing the schedule in formulation plants. That is why there has been a lot of research in this area, with different setups and applications. The majority of this research makes use of MIP. Often, they use some kind of heuristic to obtain the final schedule, since the MIP is not solvable for a realistic instance. For example, Schmidt et al. (2022) and Oujana et al. (2023) both use sequence-dependent changeover times and use two different levels of production, which is similar to our setup. They show that it is feasible to model these kinds of processes in a MIP, but also make it clear that there is still more research to do with regards to actually solving the MIPs. Ferreira et al. (2009) also describe this problem of being unable to solve the MIP formulation, but mainly focus on improving or changing the solution methods to reduce the computation time. They mention that one of the main problems that still remains is to close the integrality gap. This is the gap between the optimal solution of the linear relaxation of the model and an optimal integer solution. Finding methods to close this integrality gap is the goal of this thesis.

To be able to get into the strengthening of the model, a few definitions for MIPs and its solution space have to be given. A *polyhedron* $P \subseteq \mathbb{R}^n$ is the set of all points $x \in \mathbb{R}^n$ that satisfy a finite set of linear inequalities. For an LP, the set of feasible solutions is a polyhedron. For a MIP, we use an *integer hull*, which denotes the convex hull of all integer points. A *cutting plane* for a MIP is an inequality that cuts off part of the polyhedron of the linear relaxation of the MIP, without cutting off part of the integer hull. A *face* F of a polyhedron P is a set of points $x \in P$ that satisfy a certain valid inequality of the system with equality. A face of P is said to be *proper* if it is nonempty and properly contained in P . Inclusionwise maximal proper faces of P are called *facets*. For further used definitions from integer programming and polyhedral theory, we refer to the definitions in Chapters 2 and 3 of Conforti et al. (2014a).

Early works that introduce finding facets in the context of MIP to strengthen its formulation are by Grötschel and Padberg (1979a, 1979b). Here, facet-defining inequalities are applied to the Traveling Salesman Problem. They mention that facets are the strongest cutting planes that can be found in integer programming. Hence, in this thesis we focus on finding facet-defining inequalities.

This thesis is based on the formulation plant as seen in Yfantis et al. (2019), where a MIP is set up to model the plant. He later used an integer program (IP) (Yfantis et al., 2022) where all variables are integers or binary and include a predetermined route. This implies that there are way more variables in this model, since every possible route for all orders gets its own variable. However, both of Yfantis' models are not scalable to an example instance, so instead he used heuristics to solve the problem and plan every order one by one. This implies that the schedule depends on the sequence of orders and it is not certain that the found solution is the optimal solution for the system. So we will take the model as described in Yfantis et al. (2019) and strengthen it such that larger instances can be solved. For this, we use facet-defining inequalities and implement a separation algorithm to find these specific inequalities.

The goal of this thesis is to strengthen the MIP given by Yfantis et al. (2019) for an industrial formulation plant. First, in Chapter 2, we give the detailed setup of the industrial formulation plant with its constraints and objectives. Also, the mathematical model by Yfantis and a new proposed model are given and explained, as well as a performance comparison between the two models. In Chapter 3, a subproblem is set up including only the filling side of the plant. We then look into finding and proving facet-defining inequalities. We also create and implement a separation algorithm for this problem that is used to include specific facet-defining inequalities. After that, in Chapter 4, we check perfectness of the formulation and give an improved version of the facet-defining inequalities, as well as a separation algorithm. Finally, we give some conclusions and discuss the found results and methods.

Chapter 2

Problem description

In this chapter, we go into the exact problem that is the focus of this thesis. We discuss the setup of the plant and the constraints that are present. We also discuss some assumptions that we make, either about the setup of the plant or about the mathematical model. Next to that, we give the model made by Yfantis (Yfantis et al., 2019), with its variables, parameters and discuss its correctness. A new proposed model is given with an explanation of its constraints and then this model is compared to Yfantis' model.

2.1 Constraints and assumptions

There are some constraints and assumptions made for the industrial formulation plant that we study (Figure 1.1). We assume that the raw material going into the plant is available without restriction on time and amount. Since the materials are widely available, we can assume that there always is enough for production. Next to that, we assume that there is no loss of material during the process, so everything that goes in has to go out of the system or still be in storage within the system. An *order* is the name of both the packaged products and the raw material, as we do not differentiate between those two on the machines. We just follow an order from raw material to packaged product. We also assume that the formulation lines are used to their full capacity, so it is never the case that less than the capacity is inserted. The filling station does not have a tank to intermediately store materials, so (some of) the material will remain in either the formulation line or the buffer tank until the entire filling process is completed. For the transfer from a formulation line to a buffer tank, the amount of time it takes is negligible and thus we take this transfer to be instantaneous. Between orders, we need to take a changeover time into account. For the formulation lines, this is only the case when switching between different materials, while for the filling station, there is always some changeover time between orders. For the buffer tanks, there is no changeover time. The processing times depend on the material and the machine it is processed on, since it is possible that there are different types of machines installed over different years with different speeds. For the buffer tanks, it is only possible to fill and empty the tank from or to one station at a time. There are no precedence constraints on the orders and preemption is possible, so we are allowed to fill a bit of the product and then empty it into a buffer tank before the entire batch is done filling or to stop filling at any time when filling from the buffer. The demand per order is assumed to be known before scheduling, hence we have a deterministic model. The *makespan* is the amount of time that is needed to satisfy the demand. The objective is to minimize the makespan. The preferred time horizon for scheduling is one month.

As mentioned, we use a discrete-time model. It is stated in Floudas and Lin (2004) that for discretization, the greatest common divisor is a good measure to indicate the time intervals. That is why we

use time blocks of 1 hour, as most of the processing times in a real-life instance are multiple hours, so 1 hour is at least a common divisor. This is a sufficiently large interval that the advantages of a discrete-time model outweigh those of a continuous-time model as explained in Chapter 1.

2.1.1 Definition of sets, parameters and variables

Given the setup of the plant and the mentioned assumptions, some variables and parameters are defined. Here, we will give the general ones, while in the next subsection we will focus on Yfantis' model.

First, the sets of instances need to be defined. The orders are given by the set I . For the machines, they are given by the set of formulation lines J^{FL} , the buffers J^{ST} , and filling stations J^{FS} . Together, the sets of machines denote all machines in the system

$$J = J^{FL} \cup J^{ST} \cup J^{FS}.$$

The time slots are given by the set $\mathcal{T} = \{0, 1, \dots, T\}$.

There are both continuous and binary variables used in the model, these are explained below.

$B_{ijt} \in \mathbb{R}$ denotes the amount of order i going into buffer or filling station j at time t .

$B_{ijt}^R \in \mathbb{R}$ denotes the amount of order i being released from formulation line, buffer or filling station j at time t .

$B_{ijt}^{ST} \in \mathbb{R}$ denotes the amount of order i inside buffer j at time t .

$$W_{ijt} = \begin{cases} 1 & \text{if formulation line or buffer } j \text{ receives order } i \text{ at time } t, \\ 0 & \text{otherwise.} \end{cases}$$

$$W_{ijt}^R = \begin{cases} 1 & \text{if formulation line or buffer } j \text{ releases order } i \text{ at time } t, \\ 0 & \text{otherwise.} \end{cases}$$

$$S_{ijt} = \begin{cases} 1 & \text{if buffer } j \text{ is storing order } i \text{ at time } t, \\ 0 & \text{otherwise.} \end{cases}$$

There are also some parameters, of which the notation is given as follows:

D_i : demand for order i .

B_j : capacity of formulation line or buffer j .

F_{ij} : flow rate of order i on machine j , amount of material per time unit.

co_{isj} : changeover time going from order i to order s on machine j .

p_{ij} : processing time of a tank of order i on formulation line j .

2.2 Initial model

The model made by Yfantis (Yfantis et al., 2019) states most of the variables and some of the constraints that are used. In order to get some basic performance measure, we added the missing constraints and variables to Yfantis' model, such that the solution obtained satisfies all constraints of the plant. The entire model is too elaborate to explain here and can be found in Appendix A.1. In this subsection we will focus on the variables and sets that are specific for this model.

The set of possible processing lengths on a filling machine is denoted by $K = \{1, 2, \dots, 2 \cdot p^s + 1\}$, where p^s denotes the shift length.

Finally, the variables that are specific for Yfantis' model are:

$$X_{ijkt} = \begin{cases} 1 & \text{if filling station } j \text{ starts processing order } i \text{ at time } t \text{ that} \\ & \text{takes } p_k \text{ time units to complete processing,} \\ 0 & \text{otherwise.} \end{cases}$$

Here, the parameter $p_k \in \{0, \dots, 2 \cdot p^s\}$ denotes the processing length.

$Y_{ijt} \in \mathbb{R}$ denotes the units of time that order i processes at filling station j starting at time t .

All constraints (1b) to (1y) of this model are presented in Appendix A.1.

2.2.1 Incorrectness of the model

In Figure 2.1a an example of a schedule for Yfantis' model is given. There are some gaps in the schedule that do not seem necessary. This implies that some of the constraints used in Yfantis' model are not correct. The incorrect constraints are Inequalities (1o), (1p), (1q) and (1r). Inequality (1o) is also given here:

$$X_{ijkt} + \sum_{r \in I} \sum_{s \in K} \sum_{u=t}^{t+p_k+co_{irj}-1} X_{rjsu} \leq 1, \quad \forall i \in I, \forall j \in J^{FS}, \forall k \in K, \forall t \in \mathcal{T}. \quad (1o)$$

This constraint is supposed to model that if an order i starts processing on filling station j at time t , another order cannot start processing on that filling station within the time that order i is processing plus the changeover time from i to the new order. However, if for some $i \in I, j \in J^{FS}, k \in K, t \in \mathcal{T}$

this first part $X_{ijkt} = 0$, we still enforce that the entire sum $\sum_{r \in I} \sum_{s \in K} \sum_{u=t}^{t+p_k+co_{irj}-1} X_{rjsu}$ can be at most 1.

Say we have $p_k = 9$, $co_{i'i'j} = 3$ and for order i' , say it starts at time t with $p_{k'} = 1$. Then we have that another order of i' cannot start for the next $9 + 3 - 1 = 11$ time units because of this constraint. So the constraint induces several variables to have a joint sum of 1 or less, while in fact only if $X_{ijkt} = 1$, we should have that this set of variables is at most 1, since then indeed the machine will be occupied for this time interval. That is why we need to replace these constraints by other constraints.

Consequently, we developed a new model, given in Appendix A.2, that does not use the aforementioned constraints. The changes that are made are as follows. Since the constraints use large groups of variables, we divide it into smaller groups that do not conflict each other, regardless of which variable is 1. This is done by splitting up the sum in the constraint, so instead of summing over times and orders, we create a constraint for every one of these times and orders. This ensures that if $X_{ijkt} = 1$, still all other combinations have to be zero, while if it is 0, all separate variables are not constrained. This way, we create extra constraints, as instead of one large constraint, we now have a constraint for every time and order included. The new constraints are:

$$X_{ijkt} + \sum_{k' \in K} X_{i'jk't'} \leq 1, \quad \forall i, i' \in I, \forall j \in J^{FS}, \forall k \in K, \forall t \in \mathcal{T}, \forall t' \in \{t+1, t+p_k+co_{i'i'j}-1\}. \quad (2o)$$

The other mentioned constraints have been altered in the same way and can be seen in Inequalities (2p), (2q) and (2r).

Next to this, the objective is also changed, since only the maximum completion time was used in Yfantis' model, but that causes the model to schedule jobs later than they could be scheduled as they do not influence the objective. That is why the sum over processing times on the filling machines is

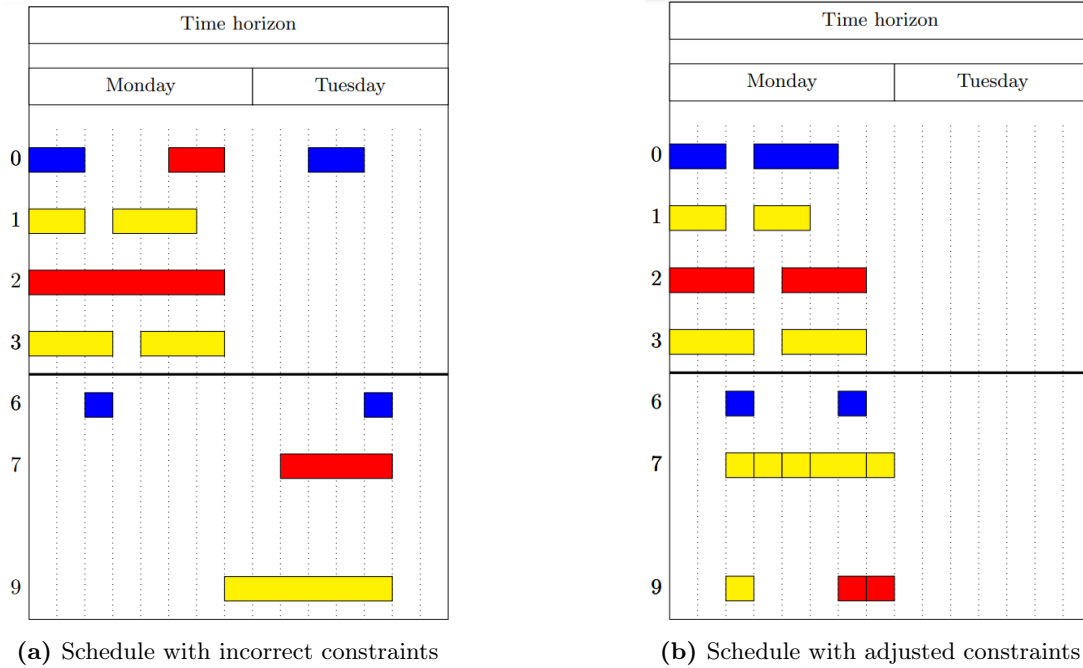


Figure 2.1: Schedule (a) without possibility to have multiple processes shortly after each other on filling stations (machines 6, 7, 9) versus schedule (b) where it is possible to have consecutive processes on filling stations.

added, multiplied with a small factor $\epsilon > 0$, to ensure jobs are finished as soon as possible, while still achieving the optimal makespan.

In Figure 2.1b the result of the new model for the same input is given. It is clear that the new model gives a better schedule with a makespan of 8 versus the old makespan of 13.

2.3 New proposed model

The number of variables in both formulations of Yfantis (Appendices A.1 and A.2) increases quickly when extending the application instance. This is mainly due to the X-variables, which have four indices, so when increasing the instance, these variables will blow up. If the precision of the discretization is increased by n , the amount of X-variables grows with $n \cdot m$, where m is given by the number of orders times the number of machines times twice the time units of a shift, which is then also increased by n as the time units in a shift depends on the time discretization. So the number of variables grows quadratically if we increase the time discretization. To improve the performance of this model, another model that contains fewer variables is created. This new proposed model is broken down in the coming sections and the complete model can be found in Appendix A.3. The improved Yfantis model (Appendix A.2) and new proposed model are then implemented and solved with Gurobi Optimizer version 10.0.3 (Gurobi Optimization, LLC, 2023) for some small instances. A comparison between these models can be found in Section 2.4.

2.3.1 Explanation of variables

In this new model, we get rid of the index k that is used in Yfantis' model. Instead of using Y_{ijt} as a continuous variable that indicates the filling time, we introduce a variable Z_{ijt} that indicates if a filling machine j is occupied at time t with order i , with X_{ijt} denoting the start of that filling process and

Y_{ijt} denoting the end. All of these variables are binary variables. To implement this, some constraints are added, while the ones containing the continuous Y_{ijt} are removed. The new and adjusted variables are given below:

$$X_{ijt} = \begin{cases} 1 & \text{if filling station } j \text{ starts processing order } i \text{ at time } t, \\ 0 & \text{else.} \end{cases}$$

$$Z_{ijt} = \begin{cases} 1 & \text{if filling station } j \text{ is processing order } i \text{ at time } t, \\ 0 & \text{else.} \end{cases}$$

$$Y_{ijt} = \begin{cases} 1 & \text{if filling station } j \text{ finishes processing order } i \text{ at time } t, \\ 0 & \text{else.} \end{cases}$$

2.3.2 Explanation of the constraints

There are several constraints in this model, to give an overview of what every constraint does we will explain each one. To enhance clarity, the constraints are split up into groups with different focus.

Objective and demand constraint

$$\min \quad C_{max} + \varepsilon \cdot \sum_{i \in I} \sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} Z_{ijt} \cdot t. \quad (3a)$$

The objective is to minimize the makespan and have every order finished as early as possible within this makespan. To ensure that the makespan will not be compromised by the second part, $\varepsilon > 0$ is a very small value.

$$\sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} B_{ijt} \geq D_i, \quad \forall i \in I. \quad (3b)$$

Constraint (3b) ensures that the demand for every order is satisfied.

Capacity constraints

$$B_{ijt}^{ST} \leq B_j, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T}. \quad (3d)$$

$$B_{ijt}^R \leq B_{ijt}^{ST}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T}. \quad (3e)$$

Constraints (3d) and (3e) ensure that the buffer does not overflow and that the amount released from the buffer does not exceed the amount inside the buffer at that moment.

Flow constraints

$$\sum_{j \in J^{FL}} B_{ijt}^R + \sum_{l \in J^{ST}} B_{ilt}^R = \sum_{m \in J^{FS} \cup J^{ST}} B_{imt}, \quad \forall i \in I, \forall t \in \mathcal{T}. \quad (3c)$$

Constraint (3c) ensures that everything that is released from the formulation lines and buffers goes into a filling station or buffer, so no material is lost in the process.

$$\sum_{u=t}^T B_{ijt} \leq \sum_{u=t}^T Z_{iju} \cdot F_{ij}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}. \quad (3g)$$

Constraint (3g) makes sure that at every time, everything that will still enter the filling station will be processed.

$$B_{ijt+1}^{ST} = B_{ijt}^{ST} + B_{ijt} - B_{ijt}^R, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \setminus \{T\}. \quad (3x)$$

Constraint (3x) ensures that the amount of material in the buffer equals the old amount, plus everything that went in at that time, minus everything that went out at that time.

Timing constraints

$$\sum_{u=t-g}^t B_{iju} - \sum_{u=t-g}^{t-1} Z_{iju} \cdot F_{ij} \leq M \cdot (1 - Y_{ijt}),$$

$$\forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \forall g \in \{0, \dots, \left\lceil \frac{\max_{k \in J^{ST}}(B_k)}{F_{ij}} \right\rceil\}. \quad (3i)$$

Constraint (3i) ensures that a filling process can only finish if all material that went in the filling station is processed. We sum over different time spans, as we know that at most the capacity of the largest buffer can go in, which takes $\left\lceil \frac{\max_{k \in J^{ST}}(B_k)}{F_{ij}} \right\rceil$ time units of processing for filling.

$$Z_{ijt} + X_{sju} \leq 1, \quad \forall i, s \neq i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}, \forall u \in \{t, \dots, t + co_{isj}\}. \quad (3j)$$

Constraint (3j) makes sure that a new filling process cannot start on a machine before the previous one is finished plus the changeover time.

$$W_{ilt}^R + X_{ijt} + \sum_{v=u}^{u+co_{isj}} W_{slv} \leq 2 + Y_{iju},$$

$$\forall i, s \in I, \forall j \in J^{FS}, \forall l \in J^{ST} \cup J^{FL}, \forall t \in \mathcal{T}, \forall u \in \{t, \dots, t + \left\lceil \frac{B_l}{F_{ij}} \right\rceil - 1\}. \quad (3k)$$

Constraint (3k) ensures that another process on the formulation line cannot start before it has finished processing on the filling station plus the changeover time.

$$W_{ijt} + \sum_{u=t}^{t+p_{ij}-1} W_{iju}^R \leq 1, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T}. \quad (3p)$$

Constraint (3p) ensures that the formulation line does not finish before it is done processing.

$$W_{ijt}^R + W_{ilt} + W_{sju} \leq 2,$$

$$\forall i, s \neq i \in I, \forall j \in J^{FL}, \forall l \in J^{ST}, \forall t \in \mathcal{T}, \forall u \in \{t, \dots, t + co_{isj}\}. \quad (3s)$$

Constraint (3s) makes sure that the changeover time is applied when emptying the formulation line into a buffer.

Coupling constraints

$$B_{ijt}^{ST} \leq M \cdot S_{ijt}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T}. \quad (3f)$$

Constraint (3f) ensures that when order i is in the buffer, the indicator for that is 1.

$$\sum_{u=1}^{t-1} Y_{iju} \leq \sum_{u=1}^{t-1} X_{iju}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}. \quad (3h)$$

Constraint (3h) ensures that at every time, at least as many filling processes have to have been started as finished.

$$Z_{ijt} \cdot (t + 1) \leq C_{max}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}. \quad (3l)$$

Constraint (3l) ensures that the makespan is at least as big as the time that a filling station is still occupied.

$$X_{ijt} \leq Z_{ijt}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}. \quad (3m)$$

Constraint (3m) ensures that if a filling process starts, the machine also has to be occupied.

$$W_{ijt}^R \cdot t \leq C_{max}, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T}. \quad (3n)$$

Constraint (3n) ensures that the makespan is at least as big as the time that a formulation line is still occupied.

$$\sum_{s=1}^{t-1} W_{ijs}^R \leq \sum_{u=1}^{t-1} W_{iju} \leq \sum_{v=1}^{t-1} W_{ijv}^R + 1, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T}. \quad (3o)$$

Constraint (3o) ensures that for all formulation lines, at all times, the amount of times a process has started is the amount of times a process has finished or one more than the amount of times a process has finished.

$$\sum_{r \in I} \sum_{u=1}^{t-1} W_{rju} - \sum_{r \in I} \sum_{u=1}^{t-1} W_{rju}^R \leq 1 - W_{ijt}, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T}. \quad (3q)$$

Constraint (3q) ensures that a new process on a formulation line can only start when no other process is busy on that line.

$$\sum_{u=1}^{t-1} B_{iju}^R + W_{ijt}^R \cdot B_j = \sum_{u=1}^t B_{iju}^R, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T}. \quad (3r)$$

Constraint (3r) ensures that at every moment in time, the total amount of material released from a formulation line is equal to the previous amount, with possibly the capacity of the machine added.

$$B_{ijt} \leq M \cdot W_{ijt}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T}. \quad (3t)$$

Constraint (3t) ensures that the indicator variable becomes 1 if something goes into the buffer.

$$B_{ijt}^R \leq M \cdot W_{ijt}^R, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T}. \quad (3u)$$

Constraint (3u) ensures that if something gets released, the indicator variable for that becomes 1.

$$B_{ijt} \leq M \cdot X_{ijt}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}. \quad (3v)$$

Constraint (3v) ensures that if something goes into a filling station, a filling process on that station has to start.

$$X_{ijt} \leq M \cdot B_{ijt}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}. \quad (3w)$$

Constraint (3w) ensures that some amount of material has gone into the filling station if a filling process starts.

$$Z_{ij,t+1} \leq X_{ij,t+1} + Z_{ijt}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \setminus \{T\}. \quad (3y)$$

Constraint (3y) ensures that if a filling station is occupied at some time, that means it either has to start at that time or it was occupied the time before.

$$Z_{ijt} - Z_{ij,t+1} + X_{ij,t+1} \leq Y_{ij,t+1}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \setminus \{T\}. \quad (3z)$$

Constraint (3z) ensures that a filling process finishes at time $t + 1$ either if it was occupied at time t and not time $t + 1$, or if it was occupied at time t and started another process at time $t + 1$.

$$\sum_{t \in \mathcal{T}} W_{ijt} = \sum_{t \in \mathcal{T}} W_{ijt}^R, \quad \forall i \in I, \forall j \in J^{FL}. \quad (3aa)$$

Constraint (3aa) ensures that material has to be released from a formulation line as often as it starts processing.

$$\sum_{u \in \mathcal{T}} X_{iju} = \sum_{u \in \mathcal{T}} Y_{iju}, \quad \forall i \in I, \forall j \in J^{FS}. \quad (3ab)$$

Constraint (3ab) ensures that total the amount of times that a filling process starts has to be equal to the total amount of times that it ends.

$$\sum_{i \in I} S_{ijt} \leq 1, \quad \forall j \in J^{ST}, \forall t \in \mathcal{T}. \quad (3ac)$$

Constraint (3ac) ensures that at most one order can be in a buffer at any given time.

$$\sum_{i \in I} Z_{ijt} \leq 1, \quad \forall j \in J^{FS}, \forall t \in \mathcal{T}. \quad (3ad)$$

Constraint (3ad) ensures that at most one order can be in a filling station at any given time.

Initialization and termination

The following constraints are not included in the model in Appendix A.3. This is because they are only the initialization and termination constraints, hence they make sure the modelling is correct and the solver will not cheat this.

$$B_{ij0}^{ST} = 0, \quad \forall i \in I, \forall j \in J^{ST}.$$

At the start the buffers are empty.

$$Z_{ij0} = X_{ij0}, \quad \forall i \in I, \forall j \in J^{FS}.$$

The filling process cannot start at time 0.

$$X_{iT} = 0, \quad \forall i \in I, \forall j \in J^{FS}.$$

The filling process cannot start at the final time point in the time horizon.

2.4 Comparison of both models

To show that the new proposed model (Appendix A.3) is indeed smaller and faster than the fixed Yfantis model (Appendix A.2) we run multiple instances on both models and compare their number of variables, number of constraints and solution time.

Table 2.1: Comparison between Yfantis' and new proposed model for an example instance with 6 orders, 13 machines (5 formulation lines, 3 buffer tanks and 5 filling stations) and 15 time slots.

	Yfantis' model	New proposed model	Ratio
# Variables	7471	5671	1.32
# Constraints	222426	50692	4.39
Solution time [s]	341.87	126	2.71

An example for one instance with 6 orders, 13 machines and 15 time slots is given in Table 2.1. Here, the results of both models are given, as well as the ratio between the two. To compare multiple instances for both models, eight different instances are given in Table 2.2. For every instance, the number of orders, machines and time slots is given, as well as the random seed. The random seed determines the numerical values of the instance, so for example the processing times or changeover times. The ratios for both models are also given in this table. In Appendix A.4 the complete tables are given for all eight instances.

Table 2.2: Ratio between number of variables and constraints and the solution time for Yfantis' and new proposed model for different instances with their number of orders, machines and time slots and random seed.

Instance	1	2	3	4	5	6	7	8
# Orders	3	3	6	3	3	3	6	6
# Machines	10	10	13	10	10	10	10	13
# Time slots	15	15	15	40	40	40	40	40
Random seed	3	74	74	74	999	3	74	74
# Variables	1.33	1.33	1.32	1.50	1.50	1.50	1.50	1.48
# Constraints	5.06	5.38	4.39	8.53	7.48	8.43	7.28	6.92
Solution time	2.11	3.87	2.71	8.17	2.51	6.54	1.39	1.42

It is clear that the new model is smaller in terms of variables and constraints and faster in solution time for every given instance. We do see that the ratio is not necessarily increasing when the instance is increased. This is not very surprising, as the randomness of the input data can cause the model to be solved slower or quicker. That is apparent if we compare the solution time of instances of the same size, but with different random seeds. These are quite different from each other. That also shows that it is hard to predict how long some input will take to solve, as there is no predictable behaviour occurring.

Chapter 3

Model strengthening

The proposed model performs better than the original one, but still cannot be scaled to an instance that covers a month of production. That is where mathematical techniques to decrease the model complexity come in. We use some different techniques, of which the idea behind the method, the outcome, its proof of correctness and, if possible, the results of implementing these methods are discussed in this chapter.

3.1 Facet-defining inequalities

As mentioned, Grötschel and Padberg (1979a, 1979b) describe that finding facet-defining inequalities and using these inequalities as cutting planes strengthens a MIP. So a useful tool in reducing the solution space of a MIP is to find facet-defining inequalities for the MIP. If we find these facet-defining inequalities, we can reduce the solution space of the LP relaxation, without cutting off feasible solutions of the MIP.

3.1.1 Setting up a subproblem

We use SCIP software (Bestuzheva et al., 2022) with IPO (Walter, 2015) to find facet-defining inequalities for the model. Walter (2016) gives a description of the algorithms used in the software. We are interested in applying this software, in particular its property to compute facets of a polyhedron. The software is suitable for up to 300 variables, which is a very small instance for our model. This means only a few different instances can be used to calculate the facet-defining inequalities and most of the time these inequalities only work for that specific instance. To extract some useful information that can be applied to all instances, we need to use a smaller model that uses fewer variables. This will also ensure that the intricacy of the obtained inequalities is lower, hence easier to grasp. That is why we look at a subproblem extracted from the entire model.

To keep the complexity of the subproblem low, we assume in this case that we have one general production rate for all machines and orders: p . Then, we do not use D_i , but $d_i = \left\lceil \frac{D_i}{p} \right\rceil$, which then denotes the number of time periods order i needs to be processed. This subproblem is given by:

$$C_{max} \geq (t + 1) \cdot Z_{ijt} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3.1a)$$

$$\sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} Z_{ijt} \geq d_i \quad \forall i \in I \quad (3.1b)$$

$$\sum_{i \in I} Z_{ijt} \leq 1 \quad \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3.1c)$$

$$Z_{ijt} \in \{0, 1\} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}. \quad (3.1d)$$

We denote the integer hull of this system to be the polyhedron P .

It is especially interesting to look at the variable that links C_{max} . This is because it can be shown that a convex combination of two possible schedules in the LP relaxation of this subproblem can strongly decrease the value of C_{max} , hence this linking is not as strong as desired. To show this, take some system with one machine where all processing times are 1 and take two possible schedules for the starting time of the orders with corresponding index sequence $0, 1, \dots, n - 1$ and $n - 1, n - 2, \dots, 0$. Both these schedules have $C_{max} = n - 1 + 1 = n$. If we then take a convex combination of these two schedules, which is by definition in the convex hull of the LP relaxation of the problem, in this instance we take a half for both. Then the schedule is given by the sequence $\frac{n-1}{2}, \frac{n-1}{2}, \dots, \frac{n-1}{2}$, which has a makespan of: $C_{max} = \frac{n-1}{2} + 1 < \frac{n}{2} + 1 < n$ for large n . So we have that a solution of the LP relaxation can have a makespan that is roughly twice as small as the makespan of the optimal solution of the MIP, hence creating a large optimality gap that has to be solved. That is why we want to find facet-defining inequalities for this subproblem, in order to strengthen the coupling of C_{max} and decrease the optimality gap between the LP relaxation and the MIP.

3.1.2 Finding facet-defining inequalities

The LP relaxation of this problem has been implemented using SCIP software with IPO. The software finds facet-defining inequalities that are then added to the original problem and can improve the LP bound. This process is repeated until it finds no more inequalities that cut off the optimal solution, so we do not know if it gives all possible facet-defining inequalities. Several instances of the subproblem are generated and put in the software. Next, we analyse these solutions to generate some general formulation given below.

Let T denote the number of time periods. A lower bound L for C_{max} is calculated by taking the total demand in time periods and dividing that over the number of filling stations f : $L := \left\lceil \frac{\sum_{i \in I} d_i}{f} \right\rceil$.

We take some tuple (t_0, t_1, \dots, t_K) of times with $K \geq 1$ and corresponding machine indices $j_1, j_2, \dots, j_K \in J^{FS}$ such that

$$L - 1 = t_0 < t_1 < \dots < t_K = T.$$

Then we get as general new constraints these *slot-jumping inequalities*:

$$C_{max} \geq L + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ijs t_s}, \quad (3.2)$$

that have corresponding (t_0, \dots, t_K) with $L - 1 = t_0 < \dots < t_K = T$, $K \geq 1$ and $j_1, \dots, j_K \in J^{FS}$. We refer to these inequalities with their corresponding requirements as slot-jumping inequalities for (t, j) .

We can make a general observation about these inequalities. The nonnegativity of the Z-variables implies that adding just one inequality to a system already ensures $C_{max} \geq L$. In the theoretical instances, we assume that all possible subsets of times are used and included as constraints. We will now prove that these inequalities are valid and that they are facet-defining inequalities. An inequality is valid for a set if every point in the set satisfies that inequality (Conforti et al., 2014a).

Theorem 1. *Given $d_i \in \mathbb{R}$ for all $i \in I$, the slot-jumping inequalities are valid for polyhedron P .*

Proof. Say we have some subset $S = \{t_1, \dots, t_K\}$ of times, $t_1 < t_2 < \dots < t_K$, that lie in P . Then the slot-jumping inequality is:

$$C_{max} \geq L + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ij_s t_s}. \quad (3.3)$$

Say the last used order is some i^* , so $Z_{i^* j^* t_{i^*}} = 1$ for some $j \in J^{FS}, t \in \mathcal{T}$ and we have $Z_{ijt} = 0$ for all $i \in I, j \in J^{FS}, t > t_{i^*}$. Then we know by (3.1a): $C_{max} \geq t_{i^*} + 1$. We also know

$$L + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ij_s t_s} \leq L + t_1 - t_0 + t_2 - t_1 + \dots + t_{i^*} - t_{i^*-1} \quad (3.4a)$$

$$\leq L + t_{i^*} - t_0 \quad (3.4b)$$

$$\leq L + t_{i^*} - L + 1 \quad (3.4c)$$

$$\leq t_{i^*} + 1. \quad (3.4d)$$

So we have that $C_{max} \geq t_{i^*} + 1 \geq L + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ij_s t_s}$, hence we can conclude that the slot-jumping inequalities (3.2) have to hold for all schedules. \square

Now that we know the slot-jumping inequalities are valid for P , we want to know if they are facet-defining. If this is the case, we get closer to having a perfect formulation, as a facet is always part of the perfect formulation. In order to prove that the slot-jumping inequalities are facet-defining inequalities, we first need to give a lemma.

Lemma 2. *Given $d_i > 0$ for some $i \in I$ and given some $L \leq T$. A schedule S that satisfies the subproblem (3.1) can always be created such that $Z_{ijt} = 1$ for some $t \in \mathcal{T}, t < C_{max}, i \in I, j \in J^{FS}$ with $L \leq C_{max} \leq T$.*

Proof. Because $t < C_{max}$, we know that setting some Z-variable to 1 at time t does not violate the constraint on the makespan (3.1a). Since $d_i > 0$, we know that at least one Z-variable has to be 1 for order $i \in I$ to satisfy (3.1b), so we can just choose that $Z_{ijt} = 1$ without violating any constraints of the subproblem (3.1). \square

Theorem 3. *Given $d_i \in \mathbb{R}$ and $d_i > 0$ for all $i \in I$, the slot-jumping inequalities are facet-defining inequalities for polyhedron P .*

Proof. Let F be the face where (3.2) is satisfied with equality. Let \hat{F} be a facet of polyhedron P , containing F , so we have $F \subseteq \hat{F}$. Let \hat{F} be described by

$$\sum_{i \in I} \sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} a_{ijt} Z_{ijt} + \beta \leq \gamma C_{max} \quad (3.5)$$

for some a, β, γ .

The idea is to take some solutions Z^0 and Z^1 that satisfy F and are very similar, but differ on some points. If we take enough of these situations, we can obtain an expression for our \hat{F} to show that $F = \hat{F}$ and thus that F is a facet.

Take some Z^0 to be greedily filled, so everything is filled densely, leaving no gaps in the schedule if possible, with $C_{max}^0 = L$. We have that $(Z^0, C_{max}^0) \in F \subseteq \hat{F}$. Say $Z_{ijt}^0 = 1$ and $Z_{i'j't}^0 = 1$ for some $i, i' \in I, j, j' \in J^{FS}$ and $t \in \mathcal{T}$ (without loss of generality by Lemma 2). Now take Z^1 to be the same as Z^0 , but with $Z_{ijt}^1 = 0, Z_{i'j't}^1 = 0, Z_{ij't}^1 = 1$ and $Z_{i'j't}^1 = 1$, so we have two orders swapped at some time t . We have that $C_{max}^1 = L$ and $(Z^1, C_{max}^1) \in F \subseteq \hat{F}$. Since they are both in \hat{F} , we can fill in the Z and C_{max} values for both Z^0 and Z^1 in (3.5) and subtract them from each other. This gives:

$$\sum_{u \in I} \sum_{v \in J^{FS}} \sum_{w \in \mathcal{T}} a_{uvw} Z_{uvw}^0 + \beta - \sum_{u \in I} \sum_{v \in J^{FS}} \sum_{w \in \mathcal{T}} a_{uvw} Z_{uvw}^1 - \beta = \gamma C_{max}^0 - \gamma C_{max}^1 \quad (3.6a)$$

$$a_{ijt} + a_{i'j't} - a_{ij't} - a_{i'j't} = \gamma L - \gamma L \quad (3.6b)$$

$$a_{ijt} + a_{i'j't} - a_{ij't} - a_{i'j't} = 0, \quad (3.6c)$$

which holds for all $i, i', j, j', t < L$.

Consider (w.l.o.g. by Lemma 2) that we have a feasible Z^0 with $Z_{ijt}^0 = 1, C_{max}^0 = L$. Take Z^1 to be the same, but with $Z_{ijt}^1 = 0$ and $Z_{ij't_1}^1 = 1$, so we move that order to a later time t_1 , after C_{max}^0 . We then have $C_{max}^1 = t_1 + 1$ and again $(Z^1, C_{max}^1) \in F \subseteq \hat{F}$. Subtracting the equations for \hat{F} we obtain:

$$a_{ijt} - a_{ij't_1} = (t_1 + 1 - L)\gamma, \quad (3.7)$$

which holds for all $i, j, t < L, t_1 \geq L$.

Consider (w.l.o.g. by Lemma 2) that we have a feasible Z^0 with $Z_{ijt}^0 = 1, C_{max}^0 = L$. If $L > \frac{\sum_{i \in I} d_i}{f}$, there is at least one $Z_{i'j't'}^0 = 0$, for some $i' \in I, j' \in J^{FS}, t' < L$. Now take Z^1 the same as Z^0 , but with $Z_{i'j't'}^1 = 1$. We have $C_{max}^1 = L$ and (3.1b) is satisfied, so $(Z^1, C_{max}^1) \in F \subseteq \hat{F}$. Subtracting the equations gives:

$$-a_{i'j't'} = 0, \quad (3.8)$$

which holds for all $i', j', t' < L$.

If $L = \frac{\sum_{i \in I} d_i}{f}$ we have $Z_{ijt}^0 = 1$ for all $i, j, t < L$. Take Z^1 the same as Z^0 but with $Z_{i'j't_1}^1 = 1$ ($t_1 \geq L$), then $C_{max}^1 = t_1 + 1$ and $(Z^1, C_{max}^1) \in F \subseteq \hat{F}$. Subtracting gives:

$$-a_{i'j't_1} = L\gamma - (t_1 + 1)\gamma \implies a_{i'j't_1} = (t_1 - L + 1)\gamma, \quad (3.9)$$

which holds for all $i', j', t_1 \geq L$.

Consider some feasible Z^0 with $C_{max}^0 = L$ and we take Z^1 the same, but with some $Z_{ij't_1}^1 = 1, Z_{i'j't_2}^1 = 1$ where $L - 1 < t_1 < t_2 \leq T$. Then $C_{max}^1 = t_2 + 1$ and $(Z^1, C_{max}^1) \in F \subseteq \hat{F}$. Subtracting gives:

$$-a_{ij't_1} - a_{i'j't_2} = L\gamma - (t_2 + 1)\gamma \quad (3.10a)$$

$$a_{i'j't_2} = (t_2 + 1 - L)\gamma - a_{ij't_1} \quad (3.10b)$$

$$a_{i'j't_2} = (t_2 + 1 - L - t_1 - 1 + L)\gamma \quad (3.10c)$$

$$a_{i'j't_2} = (t_2 - t_1)\gamma, \quad (3.10d)$$

which holds for all $i, i', j, j', L - 1 < t_1 < t_2 \leq T$.

Consider some feasible Z^0 with $C_{max}^0 = L$ and take Z^1 the same, but with some $Z_{ijt_1}^1 = 1$, $Z_{i'j't_2}^1 = 1$, $Z_{i''j''t_3}^1 = 1$ where $L - 1 < t_1 < t_2 < t_3 \leq T$. Then $C_{max}^1 = t_3 + 1$ and $(Z^1, C_{max}^1) \in F \subseteq \hat{F}$. Subtracting (3.5) at equality for both gives:

$$-a_{ijt_1} - a_{i'j't_2} - a_{i''j''t_3} = L\gamma - (t_3 + 1)\gamma \quad (3.11a)$$

$$a_{i''j''t_3} = (t_3 + 1 - L)\gamma - a_{i'j't_2} - a_{ijt_1} \quad (3.11b)$$

$$a_{i''j''t_3} = (t_3 + 1 - L)\gamma - (t_2 - t_1)\gamma - (t_1 + 1 - L)\gamma \quad (3.11c)$$

$$a_{i''j''t_3} = (t_3 - t_2)\gamma, \quad (3.11d)$$

which holds for all $i, i', i'', j, j', j'', L - 1 < t_1 < t_2 < t_3 \leq T$.

So for some subset $\{1, \dots, R\}$ of times at which some machine gets used, we have $a_{ijtr} = (t_r - t_{r-1})\gamma$ for $r \in \{1, \dots, R\}$ where we take $t_0 < t_1 < \dots < t_R < T$ and if we take $t_0 = L - 1$ it also holds for $R = 1$ because it satisfies (3.9).

Take Z^0 with $C_{max}^0 = L$ and Z^1 is the same, but with $C_{max}^1 = L + b$, so it is not satisfied with equality. It then has to hold that $\sum_{i \in I} \sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} a_{ijt} Z_{ijt} + \beta < \gamma C_{max}$. So we have to have $\gamma > 0$ as the left hand side is always non-negative. Hence, we can scale the instance, which ensures $\gamma = 1$.

Now we have

$$a_{ijt} = 0 \quad \forall i, j, t < L \quad (3.12)$$

$$a_{ijtr} = (t_r - t_{r-1}) \quad \forall i, j, 1 \leq r \leq R, \quad L - 1 = t_0 < t_1 < \dots < t_R \leq T. \quad (3.13)$$

We then have to have $\beta = L$, as that is the lower bound on C_{max} and (3.12) ensures the first part is zero if $t < L$. So we have $\beta \leq C_{max}$.

The inequality we have for \hat{F} is then

$$C_{max} \geq L + \sum_{i \in I} \sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} a_{ijt} Z_{ijt} \quad (3.14a)$$

$$\geq L + \sum_{i \in I} \sum_{j \in J^{FS}} \sum_{t=1}^{L-1} a_{ijt} Z_{ijt} + \sum_{i \in I} \sum_{j \in J^{FS}} \sum_{r=1}^R a_{ijtr} Z_{ijtr} \quad (3.14b)$$

$$\geq L + \sum_{i \in I} \sum_{j \in J^{FS}} \sum_{r=1}^R (t_r - t_{r-1}) Z_{ijtr} \quad (3.14c)$$

$$\geq L + \sum_{i \in I} \sum_{r=1}^R (t_r - t_{r-1}) Z_{ij_r t_r}. \quad (3.14d)$$

The last step holds as there needs to be some machine j_r that is used at time t_r , but it does not matter if there are multiple machines used or just one, so we can replace the sum over the machines by a specific machine j_r that is occupied at t_r .

So we have found that $F = \hat{F}$, hence F is a facet and the slot-jumping inequalities are facet-defining inequalities for our polyhedron P . \square

3.2 LP bounds

We have found facet-defining inequalities for the subproblem. The number of slot-jumping inequalities that exists is quite large for larger instances, since it includes all subsets of times larger than the lower bound. So if there are T time slots and the lower bound is L , we have 2^{T-1-L} slot-jumping inequalities. That is why we want to create an algorithm that tells us which inequality to add to the system. So to check if there is an inequality that is violated by the LP relaxation solution. If there are multiple, we want to know which inequality is the strongest, so which one gives the largest violation. In other words, given an LP solution for C_{max} , L and Z_{ijt} for all $i \in I, j \in J^{FS}, t \in \mathcal{T}$, find some subset t_1, t_2, \dots, t_n

of times that maximizes $\sum_{i \in I} \sum_{s=1}^n (t_s - t_{s-1}) Z_{ij_s t_s}$ to check if $C_{max} \geq L + \sum_{i \in I} \sum_{s=1}^n (t_s - t_{s-1}) Z_{ij_s t_s}$ can be violated. Before we propose an algorithm, we will project the current problem onto a problem with only one machine, as our C_{max} is not influenced by which machine is used, only the fact that some machine is used, so instead of using all $j \in J^{FS}$, we now only take one machine j into account. For the Z variable in the LP solution, we can then take the value $\sum_{i \in I} Z_{ijt} = \sum_{i \in I} \max_{j' \in J^{FS}} Z_{ij't}$ for all $t \in \mathcal{T}$,

as our goal is to maximize $\sum_{i \in I} \sum_{s=1}^n (t_s - t_{s-1}) Z_{ij_s t_s}$. The algorithm that we propose is as follows:

Algorithm 1 Separation algorithm

Input: $C_{max} \in \mathbb{R}, L \in \{0, \dots, T\}, \sum_{i \in I} Z_{ijt} = \sum_{i \in I} \max_{j' \in J^{FS}} Z_{ij't}$ for all $t \in \mathcal{T}$

Initialize: $U = [], Z_{max} = 0, t = T + 1, t_0 = L - 1$

while $t > L$ **do**

$t = t - 1$

if $\sum_{i \in I} Z_{ijt} > Z_{max}$ **then**

$Z_{max} = \sum_{i \in I} Z_{ijt}$

add time t to the front of U

end if

end while

$n = \text{len}(U)$

add t_0 to the front of U

if $C_{max} \geq \sum_{i \in I} \sum_{s=1}^n (U[s] - U[s-1]) Z_{ij_s U[s]}$ **then**

return ‘ (Z, C_{max}) satisfies all slot-jumping inequalities’

else

return ‘ (Z, C_{max}) violates inequality: $C_{max} \geq \sum_{i \in I} \sum_{s=1}^n (U[s] - U[s-1]) Z_{ij_s U[s]}$ ’

end if

So the idea is to go through the time horizon from back to front, remembering all times that increase the current maximum value of Z . Then, using these times, we check if Inequality (3.2) is violated.

Theorem 4. *Given is an LP solution for Subproblem (3.1) with a C_{max} , L and Z_{ijt} for all $i \in I, j \in J^{FS}, t \in \mathcal{T}$. Given is $t_0 = L - 1$. Algorithm 1 gives the largest violation for Inequality (3.2) in linear time.*

Proof. The proof is pretty intuitive. The main idea of $\sum_{i \in I} \sum_{s=1}^n (t_s - t_{s-1}) Z_{ijst_s}$ is that every time slot will contribute something to the total value. So the question is, how can we ensure that every time slot t' contributes the largest amount? That is by choosing the largest value of some $\sum_{i \in I} Z_{ijt}$ that is still to come, so picking the largest value for $\sum_{i \in I} Z_{ijt}$ where $t \geq t'$. This already shows that our algorithm will always obtain the maximum, as it does exactly this. The added value of going backwards is to find this maximum in linear time $O(T-L)$. \square

This is the fastest algorithm for this problem, as faster than linear is not possible, since we will always need to inspect all Z -values.

3.2.1 Implementation of the separation algorithm

We have implemented the separation algorithm, where we use the LP solution obtained by Gurobi as input for the algorithm and add the inequality with the largest violation to the model. This process is repeated, so multiple cutting planes are added, until the solver eventually finds an optimal solution. We have implemented the algorithm on the entire improved model (Appendix A.3), which includes formulation lines and has different processing times per order and machine. These factors influence the lower bound on C_{max} . To account for this, the calculation of the parameter L must be adapted in the implementation of the algorithm. A lower bound for both the filling stations and formulation lines is calculated, using the fastest flow rate for every order. We divide the demand for every order by its fastest flow rate, then divide by the number of machines and round up. This way, we obtain the fastest time in which all orders can be processed on the formulation lines and on the filling stations. We then take the largest of these two lower bounds and add 1. This is because the other set of machines will need at least one time unit to start or finish processing. This is not a very strong lower bound, but it does guarantee that we do not cut off any feasible solutions.

To test how well the algorithm works and to check if it makes the implementation of the improved model faster, we have solved different instances. In Table 3.1, the number of orders, machines and time slots is given for each instance, including its solution time with the algorithm (A3⁺) and without the algorithm (A3).

Table 3.1: Different instances with their number of orders, machines and time slots and their solution times for both the improved model (A3) and the improved model with the separation algorithm (A3⁺).

Instance	1	2	3	4	5	6	7	8	9	10
# Orders	3	3	3	3	3	3	3	3	3	3
# Machines	7	7	7	7	10	10	10	13	13	13
# Time slots	12	18	27	36	12	18	27	12	18	27
Solution time A3 [s]	1.27	2.20	19.36	35.64	1.37	28.40	29.08	8.22	15.07	50.78
Solution time A3 ⁺ [s]	2.12	4.65	24.68	45.17	1.59	4.60	21.44	2.43	8.97	56.12

Instance	11	12	13	14	15	16	17	18	19
# Orders	3	3	3	4	4	4	5	5	5
# Machines	16	16	16	7	7	7	7	13	13
# Time slots	12	18	27	12	18	27	27	27	36
Solution time A3 [s]	5.41	8.62	17.88	8.69	4.71	11.33	71.50	45.81	2861.61
Solution time A3 ⁺ [s]	2.40	6.25	16.89	4.23	3.81	23.08	69.78	46.75	511.12

For all instances, the same optimal objective value was obtained with and without the algorithm. This is expected and desired, as otherwise it would imply that the algorithm does cut off some feasible (optimal) solution. The results for the solution time fluctuate a lot, so the algorithm does not necessarily make the model faster. One reason for this may be because of the symmetry that is present in the slot-jumping inequalities. There are multiple inequalities that use the same time slots, but with different machines. Adding only one cutting plane at a time takes too long and thus it might be worthwhile to look into immediately including these different combinations of machines. Another reason might be that after adding a few cutting planes, adding more does not influence the model that much anymore and the problem is actually weaker somewhere else in the system. Something else about the implementation that may not be optimal is the determination of the lower bound. It is quite possible that we underestimate this lower bound, which is good because it does not cut off any feasible MIP solutions, but can also imply that the inequalities we add are weaker than desired. Because of time limitations, the exact reason and solutions were not explored for this thesis.

Something else that implies how well our cutting planes work are the LP bounds. Gurobi uses branch-and-cut methods to solve the MIP. We look at the LP bounds obtained in the root node, so before it starts branching, but after cutting planes are added. This way, we can compare the LP bounds for A3 and A3⁺. For some instances, we obtained LP bounds, while for others Gurobi was able to solve the instance without needing branching at all. There were also four instances where A3 did start branching, while A3⁺ was able to find an optimal solution without branching. The LP bounds for the instances in which branching was used for both implementations are given in Table 3.2.

Table 3.2: Different instances with their number of orders, machines and time slots and their LP bounds in the root node for both the improved model (A3) and the improved model with the separation algorithm (A3⁺).

Instance	4	10	13	17	18	19
# Orders	3	3	3	5	5	5
# Machines	7	13	16	7	13	13
# Time slots	36	27	27	27	27	36
LP bound A3	5.00983	3.10151	4.00859	4.00991	3.01039	2.001
LP bound A3 ⁺	5.01031	4.00774	4.009	4.00919	4.00875	3.32335

It is apparent that A3⁺ has a higher LP bound in every instance. In some cases they are very close, while in others there is a larger difference. The fact that the LP bound is always higher for A3⁺ implies that the cutting planes that are added do strengthen the formulation. However, the solution time is not necessarily lower for A3⁺ in these instances, as we can see in Table 3.1. Again, this implies that more research into the implementation has to be done.

Chapter 4

Extended strengthening

We have found facet-defining inequalities for the given subproblem (3.1). Now, it is interesting to check if the LP relaxation of the problem is the same as the integer hull of the MIP. If this is the case, then the formulation is called perfect and every extreme point solution to the LP relaxation is also a solution for the MIP. Hence, known methods for finding an LP solution can be applied. In this chapter we will discuss the completeness of the subproblem including the slot-jumping inequalities and find new facet-defining inequalities as well as a corresponding separation algorithm.

Implementing the subproblem (3.1) with the added slot-jumping inequalities as constraints for small instances with 1 or 2 machines, shows that the integrality gap is significantly reduced. In this subproblem with one general production rate and without changeover times, the minimum makespan is always equal to the lower bound. We know from Section 3.1.2 that adding slot-jumping inequalities will ensure that the makespan is at least the lower bound, hence it is expected that the integrality gap is reduced or even closed after adding slot-jumping inequalities. However, the LP relaxation does show that that the formulation is not yet perfect, as its optimal solution contains non-binary values for Z -variables.

4.1 Disjunctive programming

We now know that our current formulation is not perfect. This implies that there are more facet-defining inequalities to be found, to further reduce the LP convex hull. Since the previously used methods mentioned in Section 3.1.2 only gave us the slot-jumping inequalities, we need to try another approach. The method we use for this is based on disjunctive programming (Balas, 1998). In Conforti et al. (2014c), the authors describe that if disjunctive programming is done for enough steps, a perfect formulation for the problem is found. We want to take multiple steps, increasing the time horizon in every step, to eventually get general inequalities that will improve the formulation. This method creates many inequalities, so to make it more computationally feasible, a simplification is done. We assume to have only one order and one machine. So the variables only depend on the time. We started with a time horizon for which we know the perfect formulation, so $T = d$ and then used this to extend the formulation by one time unit to $T = d + 1$. Here, there are two options, either a solution has $Z_T = 0$ or $Z_T = 1$. If it is 0, it belongs to the previous system (where $T = d$), while if it is 1, we know our new C_{max} , since it is defined by this Z -variable. In that case, we can use this relation between Z_T and C_{max} in the formulation. This way, we obtained two perfect formulations whose feasible solutions together constitute all solutions of the formulation with $T = d + 1$. This idea will hold every step and hence, by induction, we can find the two perfect formulations for any time horizon.

This is the part where disjunctive programming comes in, since we have two disjunctive perfect formulations for some time horizon. So we want to describe the convex hull of these disjunctive perfect formulations to find the inequalities that describe the one perfect formulation for the entire system with some time horizon T . This convex hull is given by using Z^1 as variable in the first perfect formulation, with the right hand side multiplied by λ and the second perfect formulation using Z^2 , where the right hand side is multiplied with $(1 - \lambda)$, where $0 \leq \lambda \leq 1$. To obtain one formulation with only a Z variable, we used $Z_t = Z_t^1 + Z_t^2$. After rewriting, this gave us a system of inequalities that depends on Z_t and Z_t^2 . Then, we projected out the Z_t^2 variables one by one, using Fourier-Motzkin elimination (Dantzig, 1991). This ensured that variables are projected out, but did create many additional inequalities. Finally, we obtained a new system of inequalities, where we removed all redundant inequalities to obtain only the important inequalities for this instance. We analysed these inequalities to find more general ones that hold for any time horizon. Unfortunately, after doing three steps of this method (up to $T = d + 3$), some of the obtained inequalities have an intricate structure. It includes some inequalities that have coefficients larger than 1 for C_{max} or L . That is why we stopped investigating this method and only use it as a base to find new facet-defining inequalities.

4.2 Extended facet-defining inequality

We use the insights obtained by disjunctive programming as parameters for IPO to obtain more facet-defining inequalities. Just like in Section 3.1.2, we want to obtain a general expression out of those obtained by the software. After analysing, we group these inequalities together into one new general expression.

We take some $S \subseteq \{0, \dots, L - 1\}$ with $|S| \leq T - L$. We then take some tuple (t_0, t_1, \dots, t_K) of times with $K \geq 1$ with corresponding machine indices $j_1, j_2, \dots, j_K \in J^{FS}$ such that

$$L + |S| - 1 = t_0 < t_1 < \dots < t_K = T.$$

Then we get as a general new constraint the *extended slot-jumping inequalities*:

$$C_{max} \geq L + \sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ijs t_s}. \quad (4.1)$$

This is an extension of the previously found slot-jumping inequalities (3.2). If the set S is empty, it is the same inequality. We call these inequalities with their corresponding requirements the *extended slot-jumping inequalities* for (t, j, S) . First, we want to prove that the extended slot-jumping inequalities are indeed valid for the subproblem.

Theorem 5. *Given $d_i \in \mathbb{R}$ for all $i \in I$, the extended slot-jumping inequalities are valid for polyhedron P .*

Proof. We want to show that it holds for some subset of times S , with $|S| = m$, where $m \leq T - L$. So we have $t_0 = L + m - 1$ Next to that, say we have some subset $U = \{t_1, \dots, t_n\}$ of times, $t_1 < t_2 < \dots < t_n$, that lie in P . Then the extended slot-jumping inequality is:

$$C_{max} \geq L + \sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ijs t_s}. \quad (4.2)$$

Say the last used order is some i^* , so $Z_{i^*j_i^*t_i^*} = 1$ for some $j \in J^{FS}, t \in \mathcal{T}$ and we have $Z_{ijt} = 0$ for all $i \in I, j \in J^{FS}, t > t_{i^*}$. Then we know by (3.1a) that $C_{max} \geq t_{i^*} + 1$. We also know

$$\sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ij_s, t_s} \leq t_1 - t_0 + t_2 - t_1 + \dots + t_{i^*} - t_{i^*-1} \quad (4.3a)$$

$$\leq t_{i^*} - t_0 \quad (4.3b)$$

$$\leq t_{i^*} - L - m + 1 \quad (4.3c)$$

and

$$\sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) \leq m. \quad (4.4)$$

So

$$L + \sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ij_s, t_s} \leq L + t_{i^*} - L - m + 1 + m \quad (4.5a)$$

$$\leq t_{i^*} + 1 \quad (4.5b)$$

So we have that $C_{max} \geq t_{i^*} + 1 \geq L + \sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ij_s, t_s}$, hence we can conclude that the extended slot-jumping inequalities (4.1) have to hold for all schedules. \square

We will not prove that the extended slot-jumping inequalities are facet-defining in this thesis, but we believe this can be done in the same way as Theorem 3. Since we obtained it by software that finds facet-defining inequalities, we can assume it is indeed facet-defining.

4.3 Separation algorithm

Just like with the previously found inequalities, we want to check if we can define a separation algorithm that finds the cutting planes with the largest violation. We want to find a subset S and subset of times

t_1, t_2, \dots, t_K that maximize $\sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ij_s, t_s}$, given a solution for C_{max}

and Z_{ijt} for all $i \in I, j \in J^{FS}, t \in \mathcal{T}$. That way, we can check if Inequality (4.1) can be violated.

Just like in Section 3.2, we project the problem onto one machine, choosing the maximum value:

$\sum_{i \in I} Z_{ijt} = \sum_{i \in I} \max_{j' \in J^{FS}} Z_{ij't}$ for all $t \in \mathcal{T}$. Since it is an extension of Inequality (3.2), we can use the

previously defined Algorithm 1 for all times $t \geq L$. For the times $t < L$, we sort the entries on their

$\sum_{i \in I} Z_{ijt}$ values, such that $\sum_{i \in I} Z_{ij_0} \leq \sum_{i \in I} Z_{ij_1} \leq \dots \leq \sum_{i \in I} Z_{ij_{L-1}}$. This can be done since it does not

matter for the extended slot-jumping inequality which time index we use in S , it only depends on the size of S . So we iterate over the size of the set S , adding the items from smallest to largest, so $S = \emptyset, S = \{0\}, S = \{0, 1\}, \dots, S = \{0, 1, \dots, L-1\}$.

This algorithm uses sorting on the first L entries, which has complexity $O(L \log L)$ and then iterates over all possible sizes of S , which happens in linear time. Hence, the complexity of this algorithm is $O(L \log L)$.

Algorithm 2 Extended separation algorithm

Input: $C_{max} \in \mathbb{R}$, $L \in \{0, \dots, T\}$, $\sum_{i \in I} Z_{ijt} = \sum_{i \in I} \max_{j' \in J^{FS}} Z_{ij't}$ for all $t \in \mathcal{T}$
Initialize: $U = []$, $S = []$, $V_{max} = 0$, $E = 0$
 Sort $\sum_{i \in I} Z_{ijt}$ for $t < L$ from small to large
while $|S| < T - L$ **do**
 Extend S with smallest $t \notin S$, $t < L$
 $t_0 = L + |S| - 1$
 Execute Algorithm 1 with input t_0 to obtain U
 $n = \text{len}(U)$
 if $L + \sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^n (U[s] - U[s-1]) Z_{ijsU[s]} > V_{max}$ **then**
 $V_{max} = L + \sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^n (U[s] - U[s-1]) Z_{ijsU[s]}$
 $E := S$
 $V := U$
 end if
end while
 $k = \text{len}(V)$
if $C_{max} \geq L + \sum_{t \in E} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^k (V[s] - V[s-1]) Z_{ijsV[s]}$ **then**
 return ‘ (Z, C_{max}) satisfies all extended slot-jumping inequalities’
else
 return ‘ (Z, C_{max}) violates: $C_{max} \geq L + \sum_{t \in E} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^k (V[s] - V[s-1]) Z_{ijsV[s]}$ ’
end if

We have shown that Algorithm 1 maximizes $\sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ijs t_s}$, so we know this part is maximized in Algorithm 2. We iterate over all possible sizes for subset S , where the algorithm ensures we will always first choose the index with the largest value of $(1 - \sum_{i \in I} Z_{ijt})$. So we know that Algorithm 2 will always maximize $\sum_{t \in S} (1 - \sum_{i \in I} Z_{ijt}) + \sum_{i \in I} \sum_{s=1}^K (t_s - t_{s-1}) Z_{ijs t_s}$, hence will solve the separation problem for Inequality (4.1).

Chapter 5

Conclusion and discussion

In this thesis, we created an improved version of the MIP created by Yfantis. Next, we set up a subproblem and showed that the linking of the makespan variable is not as strong as desired. We then found facet-defining inequalities for the subproblem only involving the filling lines. This is done by using IPO for finding facet-defining inequalities. This method works well to find these inequalities, but requires some manual analysis of the inequalities to find general expressions. So for more intricate models this method is not really feasible, but for a small model it is a good method to use. That being said, it is also the case that IPO can only compute facet-defining inequalities for systems with up to about 300 variables, so that would already create a problem for larger instances or models.

A linear time separation algorithm is created and proven to be correct. This algorithm finds the slot-jumping inequalities that have the largest violation of the LP relaxation and inserts those inequalities as cutting planes for the MIP. The implementation of the improved model with the separation algorithm increased the LP bound for all instances that were investigated. However, the implementation with the algorithm is still slower than we would like. This may be because it takes too long to add all cutting planes one by one and we should make use of the symmetry of machines in the slot-jumping inequalities to add multiple cutting planes at once. Next to this, it is possible that after adding a few cutting planes, the linking of the makespan is already strengthened, but there is weakness somewhere else in the model. Finally, the determination of the lower bound is something to look into. If the lower bound used in the implementation can be increased without cutting off feasible MIP solutions, the cutting planes will further strengthen the MIP. In this thesis, we round up after summing over the orders, but it would be interesting to look into first rounding the minimum processing time for each order up and then summing over them. This should hold because we cannot process multiple orders within one time slot, so rounding up per order should not create a lower bound that is too high and thus should not cut off feasible MIP solutions. More research into the implementation is needed to find out why it is not faster than the model without the algorithm and to find out if the speed can still be improved.

The improved model including the slot-jumping inequalities we found is not a perfect formulation for the problem. It is also not guaranteed that it is feasible to find a perfect formulation, since the three steps of disjunctive programming we performed already turned out to be quite intricate. The number of computations required was very large and some of the resulting inequalities contain coefficients larger than 1, so they are hard to analyse. The disjunctive program, however, was helpful when used to change the input parameters for the IPO software. This way, we were able to obtain some new facet-defining inequalities. While we have not proven facetness of the extended slot-jumping inequalities in this thesis, we believe the techniques used in the proof of Theorem 1 will work for this proof as well. The separation algorithm proposed for the extended slot-jumping inequalities is not implemented, but

will be similar to the implementation of Algorithm 1. This would be interesting to do, to see if this will improve the computation time. Especially since the algorithm runs in $O(T \log T)$ time, where T is the length of the time horizon, it can be that it does not increase the total computation time, only the LP bounds.

Bibliography

- Balas, E. (1998). Disjunctive programming: Properties of the convex hull of feasible points. *Discrete Applied Mathematics*, 89(1), 3–44. [https://doi.org/10.1016/S0166-218X\(98\)00136-X](https://doi.org/10.1016/S0166-218X(98)00136-X)
- Bestuzheva, K., Besançon, M., Chen, W.-K., Chimela, A., Donkiewicz, T., van Doornmalen, J., Eifler, L., Gaul, O., Gamrath, G., Gleixner, A., Gottwald, L., Graczyk, C., Halbig, K., Hendel, G., Hoen, A., Hojny, C., van der Hulst, R., Koch, T., Lübbecke, M., . . . Witzig, J. (2022, April). *The scip optimization suite 8.0.0* (Version 8.0.0). Zenodo. <https://doi.org/10.5281/zenodo.6414728>
- Conforti, M., Cornuéjols, G., & Zambelli, G. (2014a). *Integer Programming* (Vol. 271). Springer International Publishing. <https://doi.org/10.1007/978-3-319-11008-0>
- Conforti, M., Cornuéjols, G., & Zambelli, G. (2014b). Perfect formulations. In M. Conforti, G. Cornuéjols, & G. Zambelli (Eds.), *Integer programming* (pp. 129–194). Springer International Publishing. https://doi.org/10.1007/978-3-319-11008-0_4
- Conforti, M., Cornuéjols, G., & Zambelli, G. (2014c). Split and gomory inequalities. In M. Conforti, G. Cornuéjols, & G. Zambelli (Eds.), *Integer programming* (pp. 195–234). Springer International Publishing. https://doi.org/10.1007/978-3-319-11008-0_5
- Dantzig, G. B. (1991). Linear equation and inequality systems. In *Linear programming and extensions* (pp. 69–93). Princeton University Press. Retrieved March 4, 2024, from <http://www.jstor.org/stable/j.ctt1cx3tvg.7>
- Ferreira, D., Morabito, R., & Rangel, S. (2009). Solution approaches for the soft drink integrated production lot sizing and scheduling problem. *European Journal of Operational Research*, 196(2), 697–706. <https://doi.org/10.1016/j.ejor.2008.03.035>
- Floudas, C. A., & Lin, X. (2004). Continuous-time versus discrete-time approaches for scheduling of chemical processes: A review. *Computers & Chemical Engineering*, 28(11), 2109–2129. <https://doi.org/10.1016/j.compchemeng.2004.05.002>
- Grötschel, M., & Padberg, M. W. (1979a). On the symmetric travelling salesman problem i: Inequalities. *Mathematical Programming*, 16(1), 265–280. <https://doi.org/10.1007/BF01582116>
- Grötschel, M., & Padberg, M. W. (1979b). On the symmetric travelling salesman problem ii: Lifting theorems and facets. *Mathematical Programming*, 16(1), 281–302. <https://doi.org/10.1007/BF01582117>
- Gurobi Optimization, LLC. (2023). Gurobi Optimizer Reference Manual. <https://www.gurobi.com>
- Khachiyan, L. G. (1980). Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1), 53–72. [https://doi.org/10.1016/0041-5553\(80\)90061-0](https://doi.org/10.1016/0041-5553(80)90061-0)
- Kopanos, G. M., Puigjaner, L., & Georgiadis, M. C. (2010, January). Optimal Production Scheduling and Lot-sizing In Yoghurt Production Lines. In S. Pierucci & G. B. Ferraris (Eds.), *Computer Aided Chemical Engineering* (pp. 1153–1158, Vol. 28). Elsevier. [https://doi.org/10.1016/S1570-7946\(10\)28193-2](https://doi.org/10.1016/S1570-7946(10)28193-2)

- Oujana, S., Amodeo, L., Yalaoui, F., & Brodart, D. (2023). Mixed-Integer Linear Programming, Constraint Programming and a Novel Dedicated Heuristic for Production Scheduling in a Packaging Plant. *Applied Sciences (Switzerland)*, 13(10). <https://doi.org/10.3390/app13106003>
- Schimidt, T. M. P., Scarpin, C. T., Loch, G. V., & Schenekemberg, C. M. (2022). A two-level lot sizing and scheduling problem applied to a cosmetic industry. *Computers & Chemical Engineering*, 163, 107837. <https://doi.org/10.1016/j.compchemeng.2022.107837>
- Walter, M. (2015). *IPO* (Version 1.0). <https://github.com/discopt/ipo/>
- Walter, M. (2016). Investigating polyhedra by oracles and analyzing simple extensions of polytopes [Accepted: 2018-09-24]. <https://doi.org/10.25673/4390>
- Yfantis, V., Babskiy, A., Klanke, C., Ruskowski, M., & Engell, S. (2022). An Improved Optimization Model for Scheduling of an Industrial Formulation Plant based on Integer Linear Programming (Y. Yamashita & M. Kano, Eds.). 49, 487–492. <https://doi.org/10.1016/B978-0-323-85159-6.50081-6>
- Yfantis, V., Siwczyk, T., Lampe, M., Kloye, N., Remelhe, M., & Engell, S. (2019, January). Iterative Medium-Term Production Scheduling of an Industrial Formulation Plant. In A. A. Kiss, E. Zondervan, R. Lakerveld, & L. Özkan (Eds.), *Computer Aided Chemical Engineering* (pp. 19–24, Vol. 46). Elsevier. <https://doi.org/10.1016/B978-0-12-818634-3.50004-7>

Appendices

A.1 Yfantis' original model

minimize

$$C_{max} \tag{1a}$$

subject to

$$\sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} B_{ijt} \geq D_i, \quad \forall i \in I \tag{1b}$$

$$\sum_{j \in J^{FL}} W_{ijt} \cdot B_j + \sum_{l \in J^{ST}} B_{ilt}^R = \sum_{m \in J^{FS} \cup J^{ST}} B_{imt}, \quad \forall i \in I, \forall t \in \mathcal{T} \tag{1c}$$

$$B_{ijt}^{ST} \leq B_j, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \tag{1d}$$

$$B_{ijt}^R \leq B_{ijt}^{ST}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \tag{1e}$$

$$B_{ijt}^{ST} \leq M \cdot S_{ijt}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \tag{1f}$$

$$\frac{B_{ijt}}{F_{ij}} \leq Y_{ijt} < \frac{B_{ijt}}{F_{ij}} + 1, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \tag{1g}$$

$$\sum_{k \in K} X_{ijk}t \cdot p_k = Y_{ijt}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \tag{1h}$$

$$\sum_{k \in K} X_{ijk}t \leq 1, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \tag{1i}$$

$$X_{ijk}t \cdot (t + p_k) \leq C_{max}, \quad \forall i \in I, \forall j \in J^{FS}, \forall k \in K, \forall t \in \mathcal{T} \tag{1j}$$

$$W_{ijt}^R \cdot t \leq C_{max}, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \tag{1k}$$

$$\sum_{s=1}^t W_{ijs}^R \leq \sum_{u=1}^t W_{iju} \leq \sum_{v=1}^t W_{ijv}^R + 1, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \tag{1l}$$

$$W_{ijt} + \sum_{u=t}^{t+p_{ij}-1} W_{iju}^R \leq 1, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \tag{1m}$$

$$\sum_{r \in I} \sum_{u=1}^t W_{rju} - \sum_{r \in I} \sum_{u=1}^t W_{rju}^R \leq 1 - W_{ijt}, \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \tag{1n}$$

$$X_{ijk}t + \sum_{r \in I} \sum_{s \in K} \sum_{u=t}^{t+p_k+co_{irj}-1} X_{rsu} \leq 1, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \tag{1o}$$

$$W_{ilt}^R + X_{ijk}t + \sum_{s \in I} \sum_{u=t}^{t+p_k+co_{isl}-1} W_{slu} \leq 2, \quad \forall i \in I, \forall j \in J^{FS}, \forall l \in J^{FL} \cup J^{ST}, \tag{1p}$$

$$\forall k \in K, \forall t \in \mathcal{T}$$

$$W_{ilt}^R + X_{ijkt} + \sum_{u=t+1}^{t+p_k-1} W_{ilu}^R \leq 2 \quad \forall i \in I, \forall j \in J^{FS}, \forall l \in J^{ST}, \quad (1q)$$

$$\forall k \in K, \forall t \in \mathcal{T}$$

$$W_{ijt}^R + W_{ilt} + \sum_{r \in I} \sum_{u=t}^{t+co_{irj}-1} W_{rju} \leq 2 \quad \forall i \in I, \forall j \in J^{FL}, \forall l \in J^{ST}, \forall t \in \mathcal{T} \quad (1r)$$

$$W_{ijt}^R \cdot t \leq C_{max} \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (1s)$$

$$B_{ijt} \leq M \cdot W_{ijt} \quad \forall i \in I, \forall j \in J^{FL} \cup J^{ST}, \forall t \in \mathcal{T} \quad (1t)$$

$$B_{ijt}^R \leq M \cdot W_{ijt}^R \quad \forall i \in I, \forall j \in J^{FL} \cup J^{ST}, \forall t \in \mathcal{T} \quad (1u)$$

$$B_{ijt} \leq M \cdot \sum_{k \in K} X_{ijkt} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (1v)$$

$$B_{ijt+1}^{ST} = B_{ijt}^{ST} + B_{ijt} - B_{ijt}^R \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \setminus \{T\} \quad (1w)$$

$$\sum_{t \in \mathcal{T}} W_{ijt} = \sum_{t \in \mathcal{T}} W_{ijt}^R \quad \forall i \in I, \forall j \in J^{FL} \quad (1x)$$

$$\sum_{i \in I} S_{ijt} \leq 1 \quad \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (1y)$$

A.2 Yfantis' fixed model

minimize

$$C_{max} + \epsilon \cdot \sum_{t \in \mathcal{T}} X_{ijkt} \cdot (t + p_k) \quad (2a)$$

subject to

$$\sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} B_{ijt} \geq D_i, \quad \forall i \in I \quad (2b)$$

$$\sum_{j \in J^{FL}} W_{ijt} \cdot B_j + \sum_{l \in J^{ST}} B_{ilt}^R = \sum_{m \in J^{FS} \cup J^{ST}} B_{imt}, \quad \forall i \in I, \forall t \in \mathcal{T} \quad (2c)$$

$$B_{ijt}^{ST} \leq B_j, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (2d)$$

$$B_{ijt}^R \leq B_{ijt}^{ST}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (2e)$$

$$B_{ijt}^{ST} \leq M \cdot S_{ijt}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (2f)$$

$$\frac{B_{ijt}}{F_{ij}} \leq Y_{ijt} < \frac{B_{ijt}}{F_{ij}} + 1 \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (2g)$$

$$\sum_{k \in K} X_{ijkt} \cdot p_k = Y_{ijt} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (2h)$$

$$\sum_{i \in I} \sum_{k \in K} X_{ijkt} \leq 1 \quad \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (2i)$$

$$X_{ijkt} \cdot (t + p_k) \leq C_{max} \quad \forall i \in I, \forall j \in J^{FS}, \forall k \in K, \forall t \in \mathcal{T} \quad (2j)$$

$$W_{ijt}^R \cdot t \leq C_{max} \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (2k)$$

$$\sum_{s=1}^t W_{ijs}^R \leq \sum_{u=1}^t W_{iju} \leq \sum_{v=1}^t W_{ijv}^R + 1 \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (2l)$$

$$W_{ijt} + \sum_{u=t}^{t+p_{ij}-1} W_{iju}^R \leq 1 \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (2m)$$

$$\sum_{r \in I} \sum_{u=1}^t W_{rju} - \sum_{r \in I} \sum_{u=1}^t W_{rju}^R \leq 1 - W_{ijt} \quad \forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (2n)$$

$$X_{ijkt} + \sum_{s \in K} X_{rjsu} \leq 1 \quad \forall i, r \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}, \quad (2o)$$

$$W_{ilt}^R + X_{ijkt} + W_{slu} \leq 2 \quad \forall u \in \{t+1, \dots, t+p_k+co_{irj}-1\} \\ \forall i, s \in I, \forall j \in J^{FS}, \forall l \in J^{FL} \cup J^{ST}, \forall k \in K, \quad (2p)$$

$$W_{ilt}^R + X_{ijkt} + W_{ilu}^R \leq 2 \quad \forall t \in \mathcal{T}, \forall u \in \{t+1, \dots, t+p_k+co_{ist}-1\} \\ \forall i \in I, \forall j \in J^{FS}, \forall l \in J^{ST}, \forall k \in K, \quad (2q)$$

$$W_{ijt}^R + W_{ilt} + W_{rju} \leq 2 \quad \forall t \in \mathcal{T}, \forall u \in \{t+1, \dots, t+p_k-1\} \\ \forall i, r \in I, \forall j \in J^{FL}, \forall l \in J^{ST}, \quad (2r)$$

$$B_{ijt} \leq M \cdot W_{ijt} \quad \forall t \in \mathcal{T}, \forall u \in \{t, \dots, t+co_{irj}-1\} \\ \forall i \in I, \forall j \in J^{FL} \cup J^{ST}, \forall t \in \mathcal{T} \quad (2s)$$

$$B_{ijt}^R \leq M \cdot W_{ijt}^R \quad \forall i \in I, \forall j \in J^{FL} \cup J^{ST}, \forall t \in \mathcal{T} \quad (2t)$$

$$B_{ijt} \leq M \cdot \sum_{k \in K} X_{ijkt} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (2u)$$

$$B_{ijt+1}^{ST} = B_{ijt}^{ST} + B_{ijt} - B_{ijt}^R$$

$$\sum_{t \in \mathcal{T}} W_{ijt} = \sum_{t \in \mathcal{T}} W_{ijt}^R$$

$$\sum_{i \in I} S_{ijt} \leq 1$$

$$\forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \setminus \{T\} \quad (2v)$$

$$\forall i \in I, \forall j \in J^{FL} \quad (2w)$$

$$\forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (2x)$$

A.3 New proposed model

minimize

$$C_{max} + \epsilon \cdot \sum_{i \in I} \sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} Z_{ijt} \cdot t \quad (3a)$$

subject to

$$\sum_{j \in J^{FS}} \sum_{t \in \mathcal{T}} B_{ijt} \geq D_i, \quad \forall i \in I \quad (3b)$$

$$\sum_{j \in J^{FL}} B_{ijt}^R + \sum_{l \in J^{ST}} B_{ilt}^R = \sum_{m \in J^{FS} \cup J^{ST}} B_{imt}, \quad \forall i \in I, \forall t \in \mathcal{T} \quad (3c)$$

$$B_{ijt}^{ST} \leq B_j, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (3d)$$

$$B_{ijt}^R \leq B_{ijt}^{ST}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (3e)$$

$$B_{ijt}^{ST} \leq M \cdot S_{ijt}, \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (3f)$$

$$\sum_{u=t}^T B_{iju} \leq \sum_{u=t}^T Z_{iju} \cdot F_{ij}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3g)$$

$$\sum_{u=1}^{t-1} Y_{iju} \leq \sum_{u=1}^{t-1} X_{iju}, \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3h)$$

$$\sum_{u=t-g}^t B_{iju} - \sum_{u=t-g}^{t-1} Z_{iju} \cdot F_{ij} \leq M \cdot (1 - Y_{ijt}), \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3i)$$

$$Z_{ijt} + X_{sju} \leq 1$$

$$\forall g \in \left\{ 0, \dots, \left\lceil \frac{\max_{k \in J^{ST}} (B_k)}{F_{ij}} \right\rceil \right\} \\ \forall i, s \neq i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T}, \\ \forall u \in \{t, \dots, t + co_{isj}\} \quad (3j)$$

$$W_{ilt}^R + X_{ijt} + \sum_{v=u}^{u+co_{isl}} W_{slv} \leq 2 + Y_{iju} \\ \forall i, s \in I, \forall j \in J^{FS}, \forall l \in J^{ST} \cup J^{FL}, \quad (3k)$$

$$Z_{ijt} \cdot (t+1) \leq C_{max}$$

$$\forall t \in \mathcal{T}, \forall u \in \left\{ t, \dots, \left\lceil \frac{B_l}{F_{ij}} \right\rceil \right\} \\ \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3l)$$

$$X_{ijt} \leq Z_{ijt}$$

$$\forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3m)$$

$$W_{ijt}^R \cdot t \leq C_{max}$$

$$\forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (3n)$$

$$\sum_{s=1}^{t-1} W_{ijs}^R \leq \sum_{u=1}^{t-1} W_{iju} \leq \sum_{v=1}^{t-1} W_{ijv}^R + 1$$

$$\forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (3o)$$

$$W_{ijt} + \sum_{u=t}^{t+p_{ij}-1} W_{iju}^R \leq 1$$

$$\forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (3p)$$

$$\sum_{r \in I} \sum_{u=1}^{t-1} W_{rju} - \sum_{r \in I} \sum_{u=1}^{t-1} W_{rju}^R \leq 1 - W_{ijt}$$

$$\forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (3q)$$

$$\sum_{u=1}^{t-1} B_{iju}^R + W_{ijt}^R \cdot B_j = \sum_{u=1}^t B_{iju}^R$$

$$\forall i \in I, \forall j \in J^{FL}, \forall t \in \mathcal{T} \quad (3r)$$

$$W_{ijt}^R + W_{ilt} + W_{sju} \leq 2$$

$$\forall i, s \neq i \in I, \forall j \in J^{FL}, \forall l \in J^{ST}, \\ \forall t \in \mathcal{T}, \forall u \in \{t, \dots, t + co_{isj}\} \quad (3s)$$

$$B_{ijt} \leq M \cdot W_{ijt}$$

$$\forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (3t)$$

$$B_{ijt}^R \leq M \cdot W_{ijt}^R \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (3u)$$

$$B_{ijt} \leq M \cdot X_{ijt} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3v)$$

$$X_{ijt} \leq M \cdot B_{ijt} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3w)$$

$$B_{ijt+1}^{ST} = B_{ijt}^{ST} + B_{ijt} - B_{ijt}^R \quad \forall i \in I, \forall j \in J^{ST}, \forall t \in \mathcal{T} \setminus \{T\} \quad (3x)$$

$$Z_{ijt+1} \leq X_{ijt+1} + Z_{ijt} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \setminus \{T\} \quad (3y)$$

$$Z_{ijt} - Z_{ijt+1} + X_{ijt+1} \leq Y_{ijt+1} \quad \forall i \in I, \forall j \in J^{FS}, \forall t \in \mathcal{T} \setminus \{T\} \quad (3z)$$

$$\sum_{t \in \mathcal{T}} W_{ijt} = \sum_{t \in \mathcal{T}} W_{ijt}^R \quad \forall i \in I, \forall j \in J^{FL} \quad (3aa)$$

$$\sum_{u \in \mathcal{T}} X_{iju} = \sum_{u \in \mathcal{T}} Y_{iju} \quad \forall i \in I, \forall j \in J^{FS} \quad (3ab)$$

$$\sum_{i \in I} S_{ijt} \leq 1 \quad \forall j \in J^{ST}, \forall t \in \mathcal{T} \quad (3ac)$$

$$\sum_{i \in I} Z_{ijt} \leq 1 \quad \forall j \in J^{FS}, \forall t \in \mathcal{T} \quad (3ad)$$

A.4 Model comparison

Table A.1

Instance	1	2	3	4	5	6	7	8
# Orders	3	3	6	3	3	3	6	6
# Machines	10	10	13	10	10	10	10	13
# Time slots	15	15	15	40	40	40	40	40
Random seed	3	74	74	74	999	3	74	74

Table A.2: Instance 1

	Yfantis	Proposed	Ratio
# Variables	5881	2161	1.33
# Constraints	48115	9509	5.06
Solution time [s]	4.41	2.09	2.11

Table A.3: Instance 2

	Yfantis	Proposed	Ratio
# Variables	2881	2161	1.33
# Constraints	45910	8530	5.38
Solution time [s]	3.25	0.84	3.87

Table A.4: Instance 3

	Yfantis	Proposed	Ratio
# Variables	7471	5671	1.32
# Constraints	222426	50692	4.39
Solution time [s]	341.87	126	2.71

Table A.5: Instance 4

	Yfantis	Proposed	Ratio
# Variables	8641	5761	1.50
# Constraints	212011	24855	8.53
Solution time [s]	93.31	11.42	8.17

Table A.6: Instance 5

	Yfantis	Proposed	Ratio
# Variables	8641	5761	1.50
# Constraints	217745	29102	7.48
Solution time [s]	102.23	40.79	2.51

Table A.7: Instance 6

	Yfantis	Proposed	Ratio
# Variables	8641	5761	1.50
# Constraints	225073	26693	8.43
Solution time [s]	271.19	41.45	6.54

Table A.8: Instance 7

	Yfantis	Proposed	Ratio
# Variables	17281	11521	1.50
# Constraints	804926	110591	7.28
Solution time [s]	3.536.7	2538.76	1.39

Table A.9: Instance 8

	Yfantis	Proposed	Ratio
# Variables	22321	15121	1.48
# Constraints	1051922	152067	6.92
Solution time [s]	1497.19	1050.82	1.42