MASTER

Improving production planning and scheduling at AC

van Twist, J.

Award date:
2012

Improving Production Planning and Scheduling at AC

Master thesis

Version 1.3 (public)
March 27, 2012

Student:
J. van Twist

Supervisors:
prof. dr. ir. W.P.M. Nuijten
ir. J.W.M. Welberg

Preface

This master thesis concludes my graduation project performed during my study of Business Information Systems at Eindhoven University of Technology (TU/e). This project is the second phase of a larger research project performed in collaboration between TU/e and AC. I highly valued the contribution of several people and I would like to thank them.

First of all, I like to thank my predecessor Roel Coset, who started the first phase of the project. His clear process description and explanation of the organizational context of AC, got me going very quickly. Also his initial result and future work suggestions provided a solid foundation to build upon. I like to thank my graduation supervisor Wim Nuijten, for the guidance and confidence in the project, and for the provided feedback. I also really appreciated his effort to provide work opportunities after this project. I like to thank my supervisor at AC, Jan-Willem Welberg, for finding the time to provide me with detailed feedback and (corrections to) many essential input data. His patience, provided flexibility, and arranged meetings greatly contributed to the result of the project. Furthermore, I thank everyone else at AC who contributed to the project and/or their showed interest.

Finally I like to thank my parents for their support during the project and my friends for the great time during my study at TU/e.

Joost van Twist
Eindhoven, march 2012

Abstract

In this document, several improvements to an initially proposed solution of a planning and scheduling problem are identified. This challenging problem is found in the production process of AC[1], being the global leader in its product field. The initially proposed solution, given in the form of an automated planning tool, showed promising results of a potential cost reduction regarding the product availability for sales. However, within the original planning tool, two manual post processing steps were required to obtain a solution that was competitive with the manual production plans. Furthermore, the initial solution performed worse regarding the maintenance of stock levels and the amount of production changeovers that were required.

The found improvements mostly involve adaptations made to the original planning tool. Furthermore, many adaptations to the input data and refinements of the constraints (that restricted the planning solutions), are applied. The new performance evaluation shows an even larger cost reduction regarding the product availability for sales. Also, product stock levels are now maintained better than within the manual production plans. On top of that it follows that the two manual post processing steps, required for the original planning tool, are not required anymore.

To increase the confidence in the current planning solution several validation experiment are conducted. A drawback still present in the current solution is the increment of changeover costs in the solutions of the planning tool. Although it is the case that minimization of changeovers does not minimize the other cost objectives, the current planning tool is shown unable to find the minimum amount of changeovers. This even holds after the implementation of a post processing step that decreases the amount of changeovers generated by the planning tool. To be able to use the planning solution for the comparison of several what-if scenarios, it is shown that many evaluations of the same scenario have to be compared, in order to make reliable predictions.

[1]The name of the company is anonymized in this public version of the document, to protect confidential information.

Contents

Chapter 1

Introduction

Even though a large amount of research is conducted in the area of production planning and scheduling, still very challenging problem variants exist. Such an instance is found in the production planning and scheduling process of AC. To improve and support this process, an initial proposed solution has been suggested in the form of an automated planning tool by [4]. The work of [4] was the initial step in a larger research project by TU/e and AC as partner. This graduation project is the second phase and aims to further investigate the potential of the initially proposed solution and to possibly identify several improvements. Some of these improvements were already suggested by [4].

For more information about the organizational context of AC and the background of the planning problem, the reader is referred to the work of [4]. Also, an overview of the research done in the area of planning and scheduling is given by [4] and is extended here in Chapter 2. The planning problem at AC was mathematically formalized by [4] and found to be of large complexity. A drawback of this description is that it is based on solving a planning problem (assigning productions to fixed buckets). One potential improvement is to base the problem description on solving a scheduling problem that is more tailored towards the actual problem that is solved. This type of problem description and a recap of the textual problem description is given in Chapter 3.

The relevant optimization criteria for the suggested solution can be found in Section 3.2.4. A comparison of the model output with the manual planning by [4] suggested a potential improvement of 17% in the non-delivery costs, however with an increase of the inventory deficit costs and the setup costs. On top of that two manual post processing steps were required on the generated schedules by the model. In Chapter 5, several improvements are identified that remove the necessity of these post processing steps. Also, the initial evaluation given in Chapter 5 shows that all cost objectives are reduced compared to the original model of [4]. Furthermore, two automatic post processing steps (Section 5.6) are designed to improve the quality of the generated schedules by the model.

Other improvements involve changes to the technical constraints and fixing errors in the input data of the model. As can be seen in the updated problem description in Chapter 3, some technical constraints have been added, whereas others have been removed, compared to the initial model of [4]. Due to the large amount of changes in the model, a new implementation description of the current model is given in Chapter 4. A final evaluation study has been conducted to measure the quality of the current model in Chapter 6. This is followed by a conclusion and a suggestion for future work in Chapter 7.

Chapter 2

Literature

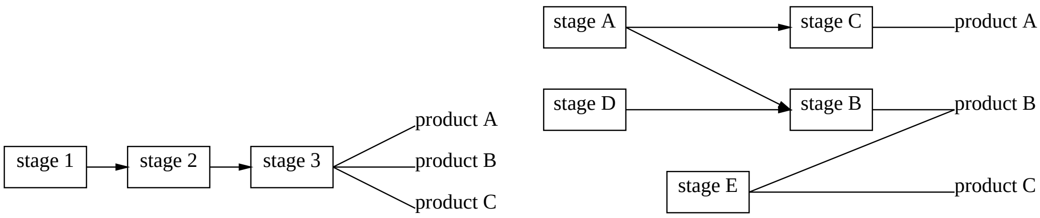
The objective of production planning is to be able to fulfill customer demand for a given set of time periods. This involves the allocation of various resources within the production facility over a time period generally covering a few weeks to a few months. This type of planning is also referred to as medium-term scheduling in literature. Long-term production planning on the other hand involves actual changes to the supply chain, such as facility locations. Short-term scheduling is in general more detailed and provides a production schedule on a short time horizon covering several days to minutes. The main issue is to decide when, where, and how to produce a set of items given a set of various processing recipes. Scheduling can have lots of different objectives such as minimizing makespan, minimizing earliness/tardiness costs and maximizing profit [8]. Since the boundaries of planning and scheduling problems are not well established and there is an intrinsic integration between these decision making stages, there is a lot of work in the literature addressing the simultaneous consideration of planning and scheduling decisions [28]. Also, batching decisions (i.e., the number and size of batches) are often treated as planning decisions (and thus provided to the scheduling problem), but can also be viewed as part of the scheduling problem [20]. Another thin boundary in literature is between the use of the terms *resource* and *unit*. A resource is usually a physical material or equipment with a certain capacity. Units on the other hand are mostly the main producing machines and can perform one task at the time. Units can be viewed as resources modeling wise (by giving them a capacity of 1) and hence the term unit is often used together with the term resource. The purpose of chapter is to give an overview of the research done in planning and scheduling in the batch process industry and hereby highlighting some of the recent developments covering the last few years.

2.1 Classification of scheduling problems

The classification of scheduling problems shows that there is a tremendous diversity of factors that must be accounted for, which makes the task of developing unified general methods quite difficult [21]. Table 2.1 gives an overview of the various characteristics. This overview shows that there are many different combinations possible of scheduling problems in practice. An important feature here is the process topology:

- Sequential processes consist either of a single stage or multiple stages. Within these type of processes, batches are used to represent production, thus it is not necessary to consider mass balances explicitly. Two different types of plants are distinguished here: Multiproduct plants produce multiple products following a sequential similar recipe. Multipurpose plants however, consist of general purpose equipment (resources) used to manufacture a variety of products, where each product can have different task structures and equipment requirements. This difference is illustrated in Figure 2.1. The scheduling for multipurpose plants is hence significantly more difficult that that of multiproduct plants [24].

- In networked processes the topology is of arbitrary structure and the network processes are a mix of convergent and divergent flow paths. Hence material balances are required to be taken into account explicitly. It involves the splitting and mixing of batches and the use of complex processing recipes.



(a) Multiproduct (b) Multipurpose

Regarding the processing tasks two types are distinguished: Batch tasks and continuous tasks. Batch tasks have a fixed duration and batch size. Furthermore the final product is produced/delivered in it's entirety at the end of the tasks execution. Continuous tasks, however, usually have a fixed processing rate and sometimes can have a minimum or maximum bound to either the processing rate or the minimum and maximum duration of the task. On top of that the products being produced are added to the stock during the tasks execution instead of at the tasks ending time.

The produced materials and products may be stored according to different policies. Unlimited intermediate storage (UIS) means that produced goods may stay in the inventory for infinite time without being consumed. If this duration is limited, because for example the goods spoil after some time, a finite intermediate storage (FIS) is applied. It might be the case that produced material require to be consumed immediately after production and then a zero wait (ZW) policy is maintained. If there is no intermitted storage at all (NIS), this means subsequent tasks that produce and consume the same material can be seen as one task.

Produced products might require to be delivered during the manufacturing process. This means that scheduling also needs to determine when to deliver which products and in what quantity, besides when and how to produce them. The demand of all products can either be at the end of the scheduling horizon or at intermediate due dates. In some cases due dates are given in terms of a window having a minimum and/or maximum delivery time. In this case earliness and tardiness costs can be introduced as well in the objective, if orders are delivered too early or too late respectively.

In the last decade a new feature of scheduling problems that gets studied, is the degree of uncertainty within production scheduling. Namely, in practise many of the parameters that are associated with scheduling are not known exactly. Parameters like raw material availability, prices, machine reliability, and market requirements vary with respect to time and are often subject to unexpected deviations [16]. The work in [16] provides an analysis on the sources of uncertainty in process scheduling and also gives an overview of the different modeling solutions dealing with these various unknown parameters. Adjusting the schedule upon realization of these uncertain parameters or occurrences of unexpected events is called reactive scheduling. The reactive scheduling corrections are performed either at or right before the execution of scheduled operations and are applied to the original (deterministically) obtained schedule.

processing topology	sequential, network
intermediate storage policy	unlimited (UIS), no intermediate storage (NIS), zero wait (ZW), finite intermediate storage (FIS)
changeovers	sequence dependent, time/frequency dependent, unit dependent, none
operation modes of processing tasks	batch, continuous
demand patterns	end of horizon, intermediate dates
resource considerations	renewable, none
objectives	minimize makespan, minimize earliness/tardiness/changover costs, maximize profit, inventory related, etc...
degree of uncertainty	deterministic, stochastic

Table 2.1: Scheduling problem characteristics

2.2 Optimization models

There exist excellent papers that review and compare the research done regarding the modeling of scheduling problems. In [7] various mixed integer linear approaches (MILP) are reviewed. This review is based on the separation between continuous and a discrete problem formulation of scheduling problems, i.e., between the way time is modeled. Another review [20] makes the main distinction between the processing topology. This latter approach is also meaningful, because all modeling approaches that are based on a sequential process topology, are batch oriented. Furthermore, only the network based topology has both discrete and continuous time modeling solutions as shown in the overview [21].

Both discrete and continuous based modeling approaches have two basic representations for the network based process topology.

- The State-Task-Network (STN) representation. The STN [14] is a directed graph consisting of two types of nodes: State nodes that represent raw materials, intermediate products and final products. Task nodes on the other hand represent the processing tasks and are related to state nodes representing the amount of material/product consumed or produced. The advantages of this representation are a clear distinction between operations and resources, the avoidance of precedence relations (they are implied by the presence of material) and the allowance of very general processing recipes [13]. On the downside, as argued by [23], all tasks can change only material states and resources are handled in an unique manner. For example to handle multiple resources that can perform the same tasks and use/produce the same materials, task duplication is required. This drawback is illustrated in Figure 2.1. Given n resources to produce product b from product a, n tasks are required to model each resource thus increasing the model size.
- Resource-Task-Network (RTN) representation. The RTN is an extension of the STN by [23]. The resource-task network process regards all processes as bipartite graphs consisting of two types of nodes: resource and tasks nodes. The concept of a resource is here very general and can consist of materials, processing equipment, storage and utilities. In contrast to the STN representation, where a task consumes

and produces materials, in a RTN, a task is assumed only to consume and produce resources. One advantage over the STN is illustrated in Figure 2.1. Since resources can be implicitly modeled in the RTN this leads to a more efficient model. A recent extension [27] of the RTN adds some new modelings features such as more realistic demand fulfillments (i.e., by introducing delivery start and end window), adding capacity bounds to resources, and extending the functionalities of a single task. The task functionality is extended by improving the interaction between tasks and resources (tasks can modify the resource capacities), and also allowing external resource transfers and delivery windows instead of fixed due dates (production orders are modeled as resources in the RTN). In the earlier RTN representation a resource node is represented using the variable $R_{r,t}$ that gives the amount of resource r at period t and using predefined maximal and minimal values to bound these resources. In the extension [27] a feature is added that allows to set the minimal and maximal values during execution of the model. This allows for scenarios that involve storage tanks for example, to be modeled more efficiently.

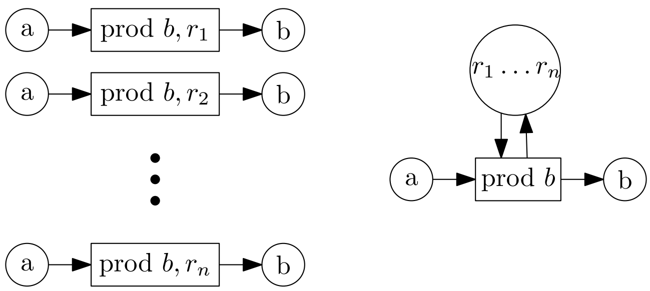


Figure 2.1: RTN versus STN representation

2.2.1 Discrete time models

In discrete time modeling approaches the time horizon is divided into fixed intervals of equal length. The start and end times of all planned events are linked to the boundaries of these intervals. The discrete formulation of scheduling problems was the first modeling technique and originated from a solution to the job shop scheduling problem [2]. The proposed modeling solution by [2] already uses a discrete time division, event sequence constraints and can deal with the integration of setup costs (i.e., changeovers).

The next series of discrete time models are based on the STN representation. An early example is the work of [14] using a MILP formulation. The key discrete variable here is W_{ijt}. W_{ijt} is a binary variable that decides whether task i starts at time interval t on unit j. Assistance variables are added to deal with batch sizes and mass balances. Optionally, other assistance variables can be added to deal with sequence-dependent changeovers. Mathematical constraints are added to enforce any (custom) feature of the scheduling problem at hand.

The advantage of the discrete time approach is that constraints are added in a relatively straightforward manner, mainly because of a fixed time grid used by each resource. The discrete time approach suffers however from two major drawbacks. First of all, the division of the time horizon into fixed intervals results in suboptimal schedules. This approximation can even lead to infeasible schedules in some cases. Namely, if too few intervals are chosen, the total number of possible timing decisions might be not be sufficient to produce a workable schedule. For example, consider an extreme situation with only one interval, if within this interval a resource is used by each event then only one event can be scheduled. With more available intervals however, the resource can be used by more events leading to a better schedule. Notice that in this case if the objective was to minimize the makespan, this problem would lead to an infeasible schedule. This is because when minimizing the makespan all events have to be scheduled. Another large drawback is that if the number of time intervals (to improve model accuracy), units and tasks increases, the number of binary variables W_{ijt} becomes very large and a lot of computational power is required to solve the problem. To tackle the last mentioned problem a few improvements were made [8]: i), a reformulation that reduces the gap between the optimal solution and its LP relaxation counterpart, ii) adding cut constraints which are redundant but reduce the region of integer infeasibility, iii) intervening in the branch and bound solution procedure, iv) the use of decomposition that divides a large and complex problem into smaller subproblems (see Section 2.2.3). Furthermore, discrete time models based on the RTN representation were developed [23]. This method only used three types of variables defining the task allocation W_{it}, the batch size B_{it}, and the resource availability R_{rt}. The batch scheduling problem is now reduced to a more simple resource balance problem carried out in each predefined time period. The unit index is removed under the assumption that it is predefined what task runs on what unit (i.e., each task runs exactly on one unit). This method leads to less variables and hence a more efficient computation, but limits the type of problems that can be modeled.

2.2.2 Continuous time models

Because of the aforementioned limitations of the discrete time models, continuous time models were introduced. The basic principle is associating the events to continuous variables instead of a fixed time grid, allowing them to take potentially any value in the time horizon eliminating the time inaccuracy factor. Like the discrete approach, continuous time models can be based on both the RTN and the STN representation. Continuous time models can be further classified into the following categories [24].

- In global event based models, the events or variable time slots are universal for all units. It is actually similar to the discrete time models except for the fact that the events can be of variable size, i.e., the timing of time intervals is treated as a model variable. Downside of this method is that the number of events/time slots still has to be set apriori, a dilemma between modeling accuracy and size [31].
- Unit-specific event based models allow each task in a unit to start at an event independent of the other units. Each unit owns a specific sequence of events. Because the unit-specific events only require an event point for the start of a task (since the events are sequential) in contradiction to the global event based models, less variables are used. However also with this method an approximation of the number of events is required.
- Within precedence based models the variables and constraints enforcing the sequential use of shared resources are modeled explicitly. Usually a variable $X_{ii'}$ explicitly stores whether batch i is processed before batch i'. The precedence based models are applied to a sequence based processing topology. This formulation allows for an intuïtive modeling of sequence dependent changeovers. Within some precedence based models the variable size directly depends on the number of batches to be scheduled resulting in larger models in practice.

For the multipurpose production variant in the continuous time models, a novel approach is given in the work of [25] using a MILP formulation. Within this approach resource diagrams (RDs) are used to represent the process structure. In a RD only task nodes and arrows between them are used. The direction of the arrows between tasks represent the task precedence just like within the STN representation. However, within RDs the arrows also represent the material that flows between the different tasks, obviating the use of material state nodes. Although the use of multiple units is presented by defining a set of allowed tasks for each unit within the RD, the use of shared resources between the various units can not be modeled unless some extension is used. A global event based formulation is given to model the RD's. The number of timeslots needs to be defined a priori. However, the work of [25] suggests a technique that starts with few timeslots and then gradually increases the amount of timeslots until no improvement is found within the objective. The method is statistically compared with two other modeling approaches. Both approaches use a continuous time STN based representation. Because of the reduction in variables and constraints in the approach of [25], the method is generally the faster approach.

A recent MILP based example of a continuous time formulation for multistage multiproduct batch plants is given in [19]. The work uses some ideas of a multipurpose batch plant formulation given by [25] to reduce the number of variables. In [19] each batch follows a series of stages ($s = 1, 2, \ldots, S$). Each stage has some units U_s available and for each batch i, the units J_i can process it. Thus the goal is to schedule for each batch i and for each stage s a single unit $j \in (U_s \cap J_i)$ in a certain time period. Two 4-index approaches are introduced:

- A 4-index unit-slot based model approach introduces for each unit some time slots ($k = 1, 2, \ldots, K_{js}$) and uses a binary variable y_{ijks} to determine whether unit j stage s processes batch i in slot k.
- A 4-index model using stage-slots uses K timeslots for each stage, hence each unit shares the same timeslots. This is in contradiction to the approach of [25] where the K timeslots are used for the entire process.

A large drawback of 4-index models is a having a large number of equations and continuous variables. Therefore, based on the best performing variations of the 4-index models and some adapted earlier approaches, the following 3-index approaches are introduced by [19]:

- A 3-index unit-slot based model uses a variable x_{iks} to determine wether batch i is processed in stage s at slot k. Another variable z_{ijs} determines wether unit j is used for batch i at stage s. The two variables are linked together by means of constraints. This leads to less variables, however more constraints are used.
- A 3-index model unit- and stage-slots based model uses an additional I contiguous stage-slots for each stage.

Although 3-index models use less binary variables more constraints and continuous variables are used. Furthermore two heuristics are introduced to reduce the solution time of both type of models. Although it is shown in [19] that their existing formulations outperform earlier methods, a clear winner does not exist between the different introduced models. Thus the authors suggest to try competitive methods for each particular scheduling problem.

Another approach for multistage multiproduct batch plants using constraint programming (CP) is given in the work [29]. It addresses several features found in industrial environments, among which sequence-dependent changeovers, topology constraints, forbidden job-equipment assignments and various objective functions. As the authors of [29] argue, constraint programming is mostly only used in hybrid methods that combine CP and MILP (see Section 2.2.4 for an example), therefore a pure CP solution is interesting to investigate. Furthermore CP has a simple declarative style and can be combined with powerful domain-specific search techniques. In [29] all the relevant constraints are described and can be augmented to any desired problem variation. A nice feature is that the scheduling of the various production orders does not need to be declared explicitly, but is enforced by means of using shared processing units. Several case studies are conducted and a decent performance is obtained by using two smart search strategies that balance the load of units at each stage. MILP formulations depend on the complexity of the various constraints. The performance of CP on the other hand depends on the implemented search strategy that is tailored towards the problem domain, as concluded by [29].

Although the discussed sequential models for multistage multiproduct batch plants have a relatively easy structure, there are several large drawbacks. First of all a preprocessing batching step (lot-sizing) is required in order to decompose the production orders into properly sized batches. In other scheduling approaches, the batching is usually integrated or done afterwards. The problem of batching beforehand is that the optimal batch quantities are not known in advance. Therefore the generated batches from the production orders may be suboptimal. Another large drawback that inventory management is significantly more difficult, i.e. the handling of inventory deficit costs. This is because also beforehand, batches have to be generated that account for stock refilling. The latter step can also vastly increase the number of batches. This is a problem, because within the sequential models, the problem size is determined by the number of batches which are to be scheduled.

2.2.3 Decomposition based solutions

Lots of literature focusses only on short-term scheduling covering a small time horizon. As shown earlier, larger discrete or continues time models can get computationally infeasible when spanning larger time-horizons. To deal with this problem, decomposition approaches are developed and are also an active point of current research. Decomposition of the planning and scheduling problems can occur at various levels. An extensive overview is given by [20] reviewing approaches for the integration of medium-term production planning and short-term scheduling.

[12] presents a novel decomposability method applied to short and medium term scheduling. The STN based method can deal with variable batch sizes and processing times, batch mixing and splitting, sequence dependent changeover times, intermediate due dates, products used as raw materials and several modes of operation. The approach extends the horizontal horizon approach presented by [18]. The horizontal horizon approach divides the entire time horizon into smaller sub-horizons, taking into account the tradeoff between demand satisfaction, unit utilization, and model complexity. First, a decomposition model is used consisting of two levels:

1. Determining the number of days in the time sub-horizon and the main products which should be included are determined. The objective is a maximal number of days in the time sub-horizon, while minimizing the model complexity. The mixed-integer nonlinear programming problem (MINLP) can be reduced to a equivalent MILP form by means of constraint rewriting. The initial given formulation is in MINLP, because a multiplication is used within the constraints that account for the model complexity limit and the production limit.

2. Adding more products to ensure a high utilization of the fist-stage processing units using a MILP formulation.

After the decomposition step, an adapted continuous scheduling approach is used to handle the short-term scheduling of the determined time horizon. This cycle is repeated until the entire time horizon is covered. In [12] a case study is conducted showing that the horizontal-horizon decomposition method is indeed an effective approach.

In [22] a hierarchical decomposition approach is presented. A three-tiered hierarchical production planning (HPP) framework for single-stage, identical parallel machines, multi-product batch plants with restricted batch size is developed. Special features are the introduction of backorders and product families. A product family is a group of products sharing the same set up features and/or aggregated demands. The HPP consists of three main levels. At the top level an Aggregate Production Planning (APP) model is used to determine the time intervals and quantities of the product families to produce. The objective is to minimize production, set-up, backorder and inventory costs. Next, at the second level, the Disaggregation Production Planning (DPP) model disaggregates the product families into actual product batch quantities, while considering the minimum and maximum batch-size requirements. The objective is to minimize the excess of production, inventory and backorder level targets that are determined by the upper level APP model. Finally, a job scheduling model (JSM) determines the assignment of jobs to production lines and the processing sequence of batches for a weekly (short) time horizon, using the outputs of the DPP model. Thus, we see that the composition is based on product family aggregation at the first and second level and horizontal decomposition at the second and last level. Although a case study by [22] shows that the approach is effective at reducing costs for the company at hand, it is unsure how it compares to other (decomposition) methods.

Another hierarchical approach is presented in [26] and introduces a global-sequence MIP formulation consisting of the following three levels: i) the selecting and sizing of batches, ii) the assignment of batches to processing units, and iii) the sequencing and timing of batches within all units. In the first level the batching of customer orders is determined, followed by the actual scheduling in the second and third level. The second and third level are merged together under the notion that there exist multiple ways to batch a product order, which impacts the scheduling. This is in contradiction to most discussed approaches that do the bathing step either together or after the first initial step. The first step determines the minimal and maximal batch sizes using plain calculations. Namely, given the maximal/minimal batch sizes possible for an order i, the maximal/minimal quantity of batches can be determined as well. These minimal and maximal batch sizes for order i can be decided by looking at the minimal/maximal batch sizes at each stage and for each unit when processing order i. A detailed explanation of this step is given in [26]. This input is fed into the second and third level where a precedence based MILP model (as discussed is Section 2.2.2) is used to solve the actual scheduling of the batches itself. The objective in the example is minimizing the lateness, earliness and processing costs, however, it can easily be extended towards any custom objectives.

The classic decomposition based method is based on defining an upper level, where an aggregate planning problem is solved defining production targets and next a lower level, where detailed scheduling problems are independently solved [1]. As argued by [5] a major drawback of this technique is that in the lower level the scheduling problems may be infeasible, because in the planning level factors like changeovers are ignored. In [5] this problem is tackled by defining an iterated version of the classic two-stage decomposition, where at the planning level an upper bound for the profit is determined. Also an estimation of changeovers is already taken into account at this level. In the lower level planning and scheduling problem products left out in the upper level are ignored, reducing the problem size. In this level a lower bound is obtained since it is a subset of the original problem.

If the difference between the lower and upper bound is beneath a certain tolerance, the procedure stops. Otherwise, the cycle is repeated after integer cuts and logic cuts are added to the MILP formulation at the upper level. The latter step reduces the overall search space. The decomposition method is compared to an original full size MILP formulation and a significant improvement was found. A downside of the proposed solution is that only one processing unit is allowed. However, another recent similar decomposition method is presented in [17] based on the STN presentation and thus eliminates the processing unit limit. They also define two levels that determine upper and lower bounds respectively and use linear cuts to reduce the problem size between iterations. Although the STN representation removes the processing unit limitation, still task duplication is required to handle multiple units when the same tasks are performed at each unit.

Another approach for removing the processing unit limitation is given in an extension of [5]. The extension [6] also uses a decomposition method based on a full-space MILP model. Contrary to [17] the solution method for single stage sequential processes is not based on the STN representation and thus unnecessary task duplication is not required. Furthermore in contrast to most of the sequential models discussed in Section 2.2.2, this MILP formulation is based on defining an amount of time slots for each unit and for some predefined time periods that are determined by the due dates or orders. This makes the approach unit-specific event based. The advantage is that now inventory levels (this also holds for [5]) can be taken into account explicitly, since the inventory levels can be monitored at each timeslot. On the downside, this method requires a postulated number of timeslots for each unit and for each time period. Too many timeslots will result in a larger model, whereas too little will lead to suboptimal solutions.

2.2.4 Other modeling solutions

The discussed modeling techniques in 2.2 are usually solved using MILP and MINLP, constraint programming (CP), or hybrid approaches where MILP and CP are integrated. An example of an approach that combines CP and MILP for both single and multiple staged sequential processes is given in the work of [9]. Here the basic strategy is two define a job assignment step solved by MILP and next a job sequencing step by using CP. The job assignment step assigns the jobs to different units. Next the job sequencing step decides whether a feasible schedule is possible. If a feasible schedule is not possible, cuts are added to the job assignment step. The general idea is that CP is a lot better at solving feasibility problems than MILP. A more recent approach [10] combines CP and MILP a similar decomposition technique called Benders decomposition. In the Benders approach a MILP master problem is solved of assigning tasks to facilities and a subproblem (solved using CP) determines the scheduling of tasks assigned to each facility. A major difference with the approach of [9] is that now a optimization problem is also considered in the subproblem formulation instead of just a feasibility problem. A drawback of this approach that it only copes with the tardiness cost of tasks and it is hard to customize the objective function.

Other used solution techniques are heuristic approaches such as simulated annealing, dispatching rules, tabu search and genetic algorithms (GA) [21]. In [15] a simulated annealing method is presented. The simulated annealing algorithm is a search algorithm, which is able to find the global minimum/maximum of an objective function in a complex search space efficiently. It does so by defining a probability acceptance function, where the acceptance of solutions depends on a temperature that is cooled down during iterations of the algorithm. At a high temperature it is more likely to accept worse solutions, however, as the temperature decreases, only better solutions are accepted. The idea of accepting worse solutions is based on the fact that a better solution may eventually be found in the future, thus avoiding getting stuck in local minima. The search space is here a feasible schedule S and it is transformed to a schedule S' by applying several neighborhood strategies.

A novel genetic approach for solving planning and scheduling is presented in [30]. In a genetic algorithm the evolution is mimicked from a random starting population and follows a series of future generations. In each generation, the quality is determined using a fitness function and the best individuals are selected for random recombination into a new population. This new population is then used in the next iteration of the algorithm. Thus, the key is define a genetic representation of the solution domain and also a fitness function to evaluate this solution domain. In [30] this representation is based on two parts: The first part defines a sequence of production order stages, by indicating the priority of each order stage with a number. The second part assigns machines (resources) to these order stages. Three different random transformations are defined, which preserve the feasibility of the solutions within the newly generated population. The method shows a slight improvement in comparison to an earlier genetic approach and is able to scale up for larger planning and scheduling problems. The fitness function can be easily adjusted to any custom objective function. However, the orders are not bound to due dates as the objective is to minimize the total makespan. Introducing this feature will introduce limitations to the random transformations and also, a new genetic representation may be required.

Chapter 3

Problem description

A thorough specification of the planning problem at AC is given in [4]. The main issue with the formulation in [4] is that an assignment problem is solved. Namely, production recipes are assigned to fixed buckets on the various production lines. In the real physical model however, tasks are scheduled at various times and are of various lengths, thus there exists a large difference between the original problem description and the actual physical model that is solved. In this chapter the mathematical problem description is revisited and a relevant textual recap is given in Section 3.1. As the scheduling problem in textual form is the same as that given in [4], some relevant parts are directly copied from [4]. In Section 3.2 a new mathematical model is given for modeling the scheduling problem at AC that is based on a MILP formulation having the advantage of incorporating a solution method directly. Because now a scheduling problem is solved this also reflects the reality better. A drawback however is that the complexity of modeling the problem increases greatly for this type of formulation.

3.1 Problem description

The main decision is to determine, for a given timeperiod, which type of product to produce on what production lines at what time, subject to many resource and technical constraints. This time period covers 3 months in general. The most important objectives are satisfying a series of product demands and keeping stock targets of each type of product at desired levels. A production line produces one product at the time and may do so by using various processing recipes, which determine the characteristics of the product and also the production throughput. The production process of product is a continuous process, because all different steps happen in a continuous chain and do not require a separate planning. As shown in [4], from the planning perspective, the production process can be seen as a black box . In this section only the relevant production context is given. For a more detailed description and an organizational context of AC the reader is referred to the work of [4].

For the planning of the product some information about the production process is relevant. In the following paragraphs relevant information about the products, changeovers and technical constraints will be explained. The changeover time is relevant, since it differs between products. Not every production line can produce every product and not all products can be produced simultaneously, which makes the information about technical constraints relevant for the production planning.

Changeovers: When a production line needs to switch from producing product x to product y a changeover takes place. Since the production is a continuous process, the production line keeps running and produces B-quality product during most of the changeovers. This B-quality product is not wasted as it is required for the production of pulp. However, the required pulp ingredient can be spun more efficient when spun directly.

Specific changeover information is considered classified and has been omitted from this version of the document. Four different changeover types have been identified, i.e. a normal change, a blocking change, a soft change, and a spinneret change. These four changeover types all have a different changeover time.

Technical constraints: These technical constrains are considered classified information and have been excluded from this version of the document.

Besides the technical constraints there are some constraints that restrict the planning process of some productions in practice regarding the scheduling of product.

1. Products *D1040 1680*, *D1015 3360* and *2100 1100* can only start from Monday till (and including) Friday.
2. Changeovers of type *spinneret* having the largest changeover time may only occur Monday till (and including) Friday.
3. During productions a minimal production time is required of 5 days. This can be more and is often less, however, the exact time depends on the recipe that has run previously on the same production line. Because of difficult modeling issues regarding this constraint a minimal period of 5 days is assumed.
4. For some types of product the production lines that are suitable for that product are further limited, due to a difference in spoolsize, or other commercial requirement. A commercial requirement for example is that some products have to obtain some chemical properties and then it can only be made on a restricted set of production lines.

3.2 Mathematical model

The classification of planning and scheduling problems, as explained in Chapter 2, implies that the production plant at AC follows a sequential, single stage, continuous, multipurpose production process. This is because each product can be produced using various processing recipes using different sets of resources. Also, at an abstract level, the intermediate products can be omitted and only one stage is needed for each product. The problem with sequential solution based models however is that a batching step is required beforehand to explicitly model production precedences. For AC this is not justified, because all productions are continuous and a batching step will lead to suboptimal schedules. Also, within the sequential models, inventory levels are difficult to manage, because there is no explicit notion of time. Therefore, it is chosen to base the new problem description on a modified version of the continuous RTN (see Section 2.2) given by [3]. As mentioned in Section 2.2.2, a continuous model has the advantage of having fewer time points as discrete models. Notice that since the timing variables are now seen as real values, it is straightforward to change the time unit from hours to days, should the problem be computationally difficult to solve in practice. Notice that the number of event points T should be approximated, just as the number of event points a certain task can overlap (Δt). Underestimation of these values will lead to suboptimal schedules, whereas an overestimation might result in computational performance issues.

The mathematical description is divided into four parts. First the inputs of the planning problem are described in Section 3.2.1. This input is a mapping from the instance of the planning problem at AC to the input of a RTN formulation. Next a description of the output is given in terms of the input in Section 3.2.2. The constraints that restrict the possible outputs are given in Section 3.2.3. Finally, the objective function defining the quality of a planning solution, is defined in Section 3.2.4.

This mathematical description leads to the following problem definition:

Definition. AC Planning and Scheduling Problem (ACPSP).

An instance of ACPSP consists of the input as given in Section 3.2.1. The most important components are the set of timeslots T, the set of tasks I, the set of resources E used by these tasks, and also, the relation between the resources and tasks given by the defined parameter $\mu_{i,e}$ (equals the capacity of resource e required by task i).

The problem is to find a solution of ACPSP (Section 3.2.2), respecting the constraints in Section 3.2.3 with a cost of at most C, i.e., for which $OF \leq C$, where OF is defined in Section 3.2.4. The most important components of the ACPSP solution are the assigment of tasks to event points ($N_{i,t,t'}$), the absolute time of each event point T_t, and the amount of materials produced by these tasks ($\xi_{i,t,t'}$).

3.2.1 Input

Time: The time unit in the mathematical model are hours, as specific production times, changeover times and constraints are all specified in hours. Instead of dividing the time horizon H into buckets of fixed size, in this formulation H is spanned by a series of events points $T = \{1, 2, \ldots, |T|\}$.

Product: There are many different products within AC. P is the set of different products at an aggregated level. A single $p \in P$ is an aggregation of a number of products at SAP's material level. A product at SAP's material level is a packaging of spools, with a specific spool size, that contains a specific weight of a specific product. In SAP there exists a forecast at this product level. For many customers, the packaging, the spool size, or the spool weight does not matter. This results in the fact that the forecasts for some different products at the material level, are forecasts for the same product at the aggregated level.

Production orders: For every product $p \in P$ there exists a forecast in tons (1000 kg) per month. This forecast is a summation of the forecasts of material numbers in SAP corresponding to $p \in P$. To cope with this, a set of production orders O is included within the model. Each production order o has a due date (due_o), product (pr_o) and quantity (dem_o). The non-delivery costs of a production order is defined as the profit of p times a modifier of the production order to emphasize strategic importance.

$ndc(o) = \mathit{profit}_{pr_o} \cdot \mathit{ndcModifier}_o$

Recipes: R is the set of different product recipes. A single $r \in R$ is the recipe to produce a single $p \in P$. A single $p \in P$ might have more than one recipe $r \in R$.

Resources: There are two different types of resources, *primary* and *secondary* resources, involved in producing products $p \in P$. The *primary* resources are the resources that actually produce the product and the *secondary* resources are resources that are constraining the primary resources with a certain capacity.

- H is the set of high-rise resources. Every $h \in H$ has a certain throughput capacity.
- L is the set of production lines. Every production line is linked to one $h \in H$.
- $S = \{D, E, F, K, P, S, X\}$ is the set of different spinneret types. Each production requires a certain type of spinneret and the total number of the same type of spinnerets in use is limited given by capacity $spCap_s$ for $(s \in S)$

Changeovers: When recipes $r \in R$ and $r' \in R$ are produced in sequence on the same production line, a changeover takes place. The time of the changeover is defined by the changeover type. There are four different types of changeovers (Section 3.1), each with its own length in hours. The parameter ttf_r, refers to the type, titer and filaments identification of the product produced by recipe r. Other more trivial parameters are found in the nomenclature section.

For changeovers the following functions exist for $r, r' \in R$:

- $cType(r, r') \in \{no, soft, normal, block, spin\}$ are the changeover type between recipes r and r'.
- $ct(r, r')$ returns the changeover time between recipes r and r'.

Current production: At $t = 0$, for all production lines $l \in L$ there exists a recipe $r \in R$ currently being produced at l. This is relevant for the possibly needed changeovers at the start of the schedule.

Tasks: The set of spinning tasks I^{spin} consists of all recipes that are possible for each production line subject to all the line/recipe constraints. These constraints are omitted in this version of the document.

$I^{\text{spin}} = \{(r, l) \in R \times L \mid \text{production line } l \text{ satisfies the classified technical constraints of } r\}$

Next the following set of changeover tasks is defined:

$I^{\text{co}} = \{i \in cType(r, r') | r, r' \in R\} \setminus \{no, soft\}$

General resources: To keep track of each resource over time all resources are generalized as specified in equation 3.1. The maximum capacities of these resources are omitted in this version of the document.

$$E = H \cup L \cup S \cup (\text{other classified technical resources}) \tag{3.1}$$

The required capacities for each task i and resource e is defined by $\mu_{i,e}$ and is as follows:

$$\mu_{i,e} = \begin{cases} tp_{r(i)} & \text{if } i \in I^{\text{spin}} \wedge e \in H \wedge hrRes_{l(i)} = e, \\ 1 & \text{if } i \in I^{\text{spin}} \wedge e \in L \wedge e = l(i), \\ 1 & \text{if } i \in I^{\text{spin}} \wedge e \in S \wedge e = spinneret_{r(i)} \\ 1 & \text{if } i \in I^{\text{co}} \wedge e \in L \wedge e = l(i), \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

3.2.2 Output

A solution of the ACPSP consists of an allocation $N_{i,t,t'}$ of tasks to event points and the amount of product $\xi_{i,t,t'}$ produced by these tasks. $N_{i,t,t'}$ is a binary variable that indicates whether task i runs from even point t to event point t'. The variable $\xi_{i,t,t'}$ determines the amount of material produced by task i from event point t to t'. The absolute time of each event point $t \in T$ is given by the variable T_t. Also we have the inventory level of each product $p \in P$ given by $I_{p,t}$ for each $t \in T$. The non-delivered amount of each production order o at event point t, is given by the slack variable $psl_{o,t}$. The binary variable $po_{o,t}$ decides whether order o is (partially) fulfilled at time t. Notice that each production order o can only be delivered on at most one event point $t \in T$. An example of the output for 3 production lines is given in Figure 3.1. The grey striped tasks represent the changeover tasks. Δt is set to 4 in the example, so all tasks can maximally overlap 4 intervals.

If a certain task $i \in I$ is excuted and $N_{i,t,t'} = 1$, this means that all resources $e \in E$ required by task i (given by $\mu_{i,e}$) should be consumed at event point t and freed at t'. Also the amount of product produced ($\xi_{i,t,t'}$) should correspond to the duration of the interval. Namely, the recipe of task i has a certain throughput and the amount produced is hence derived from the duration of the inverval divided by this throughput. If a production order $o \in O$ is delivered at time t, given by $po_{o,t}$, the correct amount should be substracted from the inventory level $I_{p,t}$ accordingly, minus the slack (undelivered amount) $psl_{o,t}$.

For example, consider task $i5$ in Figure 3.1. This task runs from event point 3 to 6 and hence $N_{i5,3,6} = 1$. Also, this means that $\forall i \in I \mid i \neq i5$, $N_{i,3,6} = 0$. The latter holds because the resource SB is consumed at $t = 3$ by task $i5$ ($\mu_{i5,SB} = 1$) and is therefore unusable by the other tasks. The amount of material produced by task $i5$ running on production line SB corresponds to the variable $\xi_{i5,3,6}$.

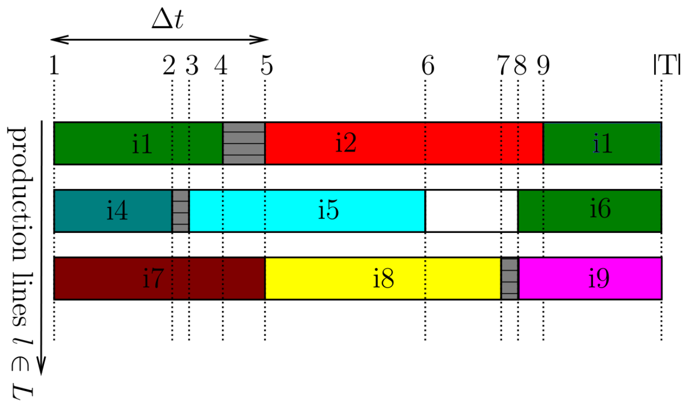


Figure 3.1: Example output for 3 production lines

3.2.3 Constraints

Timing: Constraints are needed that relate the amount of time passed between event points and the tasks that are executed on them. Constraint 3.3 states that if a certain changeover task or production task is executed, the duration of the interval is at least the changeover duration (cht_i) or the needed production time respectively. The latter is defined by the amount of material produced $\xi_{i,t,t'}$ and the throughput (tons per hour) of the recipe belonging to task i ($tph_{r(i)}$). The constraint is stated per production line $l \in L$. To check wether production line l is used by a certain task i, $\mu_{i,l}$ is used (for the production line resources $l \in L$, $\mu_{i,l}$ only takes the value 0 or 1).

$$\begin{aligned}&\forall l \in L\ \forall t,t' \in T \mid t < t' \leq t + \Delta t, t' \leq T\\&T_{t'} - T_t \geq \sum_{i \in I^{\text{co}}} (\mu_{i,l} \cdot N_{i,t,t'} \cdot cht_i)|_{t'=t+1} + \sum_{i \in I^{\text{spin}}} (\frac{\mu_{i,l} \cdot \xi_{i,t,t'}}{tph_{r(i)}})\end{aligned} \tag{3.3}$$

$$\begin{aligned}&\forall l \in L\ \forall t,t' \in T \mid t < t' \leq t + \Delta t, t' \leq T\\T_{t'} - T_t \leq & H(1 - \sum_{i \in I^{\text{co}}} (\mu_{i,l} \cdot N_{i,t,t'})|_{t'=t+1} - \sum_{i \in I^{\text{spin}}} (\mu_{i,l} \cdot N_{i,t,t'}))\\&+ \sum_{i \in I^{\text{co}}} (\mu_{i,l} \cdot N_{i,t,t'} \cdot cht_i)|_{t'=t+1} + \sum_{i \in I^{\text{spin}}} (\frac{\mu_{i,l} \cdot \xi_{i,t,t'}}{tph_{r(i)}})\end{aligned} \tag{3.4}$$

Notice that since changeover times are relatively small compared to production times, it is assumed that they can last only one time interval. Moreover, the summations of the changeovers and the production tasks can be taken together, because only one task can be executed at each interval at the same time due to the resource balance constraints. Vice versa, constraint 3.2.3 states that a maximal processing time should be accounted for. If no task can be executed between two event points, there should not be an upper bound between two event points on the amount of time passed.

Operational: Constraint 3.5 expresses that a minimal amount of material should be produced (depending on the recipe) if a spinning task is executed. The upper bound is specified a the maximal amount that can be produced if the whole time horizon was used. Notice that the minimum production time of a recipe of a task ($mpt_{r(i)}$) should be larger than zero, otherwise $N_{i,t,t'}$ can be equal to one (and thus executed), while no material is produced.

$$\begin{aligned}&\forall t,t' \in T\ i \in I^{\text{spin}} \mid t < t' \leq t + \Delta t, t' \leq T\\&mpt_{r(i)} \cdot tph_{r(i)} \cdot N_{i,t,t'} \leq \xi_{i,t,t'} \leq H \cdot tph_{r(i)} \cdot N_{i,t,t'}\end{aligned} \tag{3.5}$$

Resource balance: The variable $E_{e,t}$ maintains the amount of resource e available at event point t. This amount equals the starting amount ($E_e^{\max}|_{t=1}$)plus the amount of resources that are freed up by all previously executed tasks (those ending at t) and minus the consumption of all starting tasks (beginning at t). Constraint 3.7 states that the amount of resource can not be empty at each event point. The parameter $\mu_{i,e}$ specifies the amount of resource e that is required by task i.

$$\forall e \in E\ \forall t \in T \tag{3.6}$$

$$\begin{aligned} E_{e,t} =& E_e^{\max}|_{t=1} + E_{e,t-1}|_{t>1} + \sum_{i \in I^{\text{spin}}} \left[\sum_{\substack{t' \in T \\ t-\Delta t \geq t' < t}} \mu_{i,e} \cdot N_{i,t',t} - \sum_{\substack{t' \in T \\ t < t' \leq t+\Delta t}} \mu_{i,e} \cdot N_{i,t,t'} \right] \\ & + \sum_{i \in I^{\text{co}}} \mu_{i,e} \cdot N_{i,t,t-1} \Bigg|_{t>1} - \sum_{i \in I^{\text{co}}} \mu_{i,e} \cdot N_{i,t,t+1} \Bigg|_{t<|T|} \end{aligned}$$

$$\forall e \in E\ \forall t \in T \quad 0 \leq E_{e,t} \tag{3.7}$$

Inventory: Similar to the resource balance constraints the inventory of each product is maintained as well at each event point by the variable $I_{p,t}$. The initial inventory level for product p is defined by the parameter I_p^0. In addition, the delivery of all production orders $o \in O$ is accounted for. This might not be possible at all times, therefore a slack variable $psl_{o,t}$ is introduced that directly accounts for the unsatisfied demand for production order o (given by dem_o) at time t.

$$\forall p \in P\ \forall t \in T$$

$$I_{p,t} = I_p^0|_{t=1} + I_{p,t-1}|_{t>1} + \sum_{\substack{i \in I^{\text{spin}} \\ pr_{r(i)}=p}} \left[\sum_{\substack{t' \in T \\ t-\Delta t \geq t' < t}} \xi_{i,t',t} \right] - \sum_{o \in O | pr_o = p} dem_o \cdot po_{o,t} + psl_{o,t} \tag{3.8}$$

$$\forall p \in P\ \forall t \in T \quad I_{p,t} \geq 0 \tag{3.9}$$

Production orders: If a production order o is (partially) fulfilled at time t, represented by the binary variable $po_{o,t}$, the event point should have a ending time before or at the due date of the order. If the order is not fulfilled at all then there is no restriction on the corresponding event point.

$$\forall o \in O\ \forall t \in T \quad T_t \leq due_{o,t} + (1 - po_{o,t}) \cdot H \tag{3.10}$$

Furthermore if a production order o is not (partially) satisfied, then the sum of the slack variable $psl_{o,t}$ for all $t \in T$ should be zero. This is to prevent that the model misuses the slack variable to boost the inventory level, while no production order is satisfied.

$$\forall o \in O\ \forall t \in T \quad 0 \leq psl_{o,t} \leq dem_o \cdot po_{o,t} \tag{3.11}$$

Finally, at most one time point can be used to satisfy each production order $o \in O$.

$$\forall o \in O \quad \sum_{t \in T} po_{o,t} \leq 1 \tag{3.12}$$

Changeovers: To deal with the various changovers for each production line $l \in L$, all pairs of tasks that require a changeover should execute a changeover task (constraint 3.13). To this end all pairs of recipes $r, r' \in R$ that require a changeover and all ordered event points $t1, t2 \in T \mid t1 < t2$ are considered. The constraint states that between event point $t1$ and event point $t2$ a changeover task must be executed if there is a task executed that ends on $t1$ and has recipe r, and there is a task started at $t2$ that has recipe r', and finally, between event points $t1$ and $t2$ no other task is executed.

$$\forall l \in L\ \forall r, r' \in R\ \forall t1, t2 \in T\ \forall i \in I^{\text{co}} \mid t1 < t2, i' = cType(r, r'), i' \notin \{soft, no\}$$

$$\sum_{\substack{t \in T \\ t1 \leq t < t2}} \mu_{i',l} \cdot N_{i',t,t+1} \geq \sum_{\substack{t \in T \\ t-\Delta t \leq t < t_1}} \sum_{\substack{i \in I^{\text{spin}} \\ r(i)=r}} \mu_{i,l} \cdot N_{i,t,t1} + \sum_{\substack{t \in T \\ t2 \leq t < t2+\Delta t}} \sum_{\substack{i \in I^{\text{spin}} \\ r(i)=r'}} \mu_{i,l} \cdot N_{i,t2,t} \quad - \left[\sum_{\substack{t \in T \\ t1 \leq t < t2}} \sum_{\substack{t' \in T \\ t1 < t' < t+\Delta t}} \sum_{i \in I^{\text{spin}}} \mu_{i,l} \cdot N_{i,t,t'} \right] - \quad 1 \tag{3.13}$$

Remaining constraints: Some product types may not be produced simultaneously.

$$\forall t \in T\ \forall r, r' \in R \mid \neg allowSim(r, r')$$
$$\sum_{\substack{u,u' \in T \\ u'-u \leq \Delta t \\ u > t-\Delta t \\ u' \geq t+1}} \sum_{i \in I \mid r_i = r} N_{i,u,u'} + \sum_{\substack{u,u' \in T \\ u'-u \leq \Delta t \\ u > t-\Delta t \\ u' \geq t+1}} \sum_{i \in I \mid r_i = r'} N_{i,u,u'} < 2 \tag{3.14}$$

3.2.4 Objectives

The objective function of ACPSP is a combination of four different objectives with different weights. The best output corresponds to the minimum of this fuction.

$OF = w_{\text{ndc}} \times O_{\text{ndc}} + w_{\text{st}} \times O_{\text{st}} + w_{\text{pc}} \times O_{\text{pc}}$

- *Non-delivery costs:* One objective of a solution to ACPSP is to minimize the difference between the production need and the actual production. This difference is measured as non-delivery costs, i.e. the money that would have been earned if the product was produced. The non-delivered amount for each production order $o \in O$ is equal to the demand of the order dem_o minus any amount that might have been deliverd at a (single) time point. This is multiplied by the non-delivery costs of the order ($ndc(o)$).

$$O_{\text{ndc}} = \sum_{p \in P} \sum_{o \in O \mid pr_o = p} \left[dem_o - \sum_{t \in T} po_{o,t} \cdot dem_o - psl_{o,t} \right] \cdot ndc(o)$$

- *Stock target:* In order to account for uncertainties in the forecast, AC tries to keep the stock levels of all the products at a defined target. A ACPSP solution should keep all the stocks above their required target. If the stock is beneath its target, the solution should be penalized for that. The stock target of a product is defined in terms of days of sales corresponding to a certain amount in tons.

$$O_{\mathrm{st}} = \sum_{p \in P} \sum_{t \in T} stc_p \cdot \frac{((stockTarget_p - I_{p,t}) \uparrow 0) \cdot stDays_p}{stockTarget_p}$$

  The function O_{st} corresponds to the sum of the stock deficit costs times the number of days the stock is below the target, per event point per product. Notice this is an approximation of the actual stock deficit costs as these costs are now only evaluated at each event point. It is possible to be more precise by introducing a non linear objective function (as shown in [3]) that compares the inventory level difference and the time difference between subsequent event points.

- *Production costs:* When the complete demand can be fulfilled and all the stocks are above their required target, the next objective is minimizing the production costs.

$$O_{\mathrm{pc}} = \sum_{i \in I^{\mathrm{spin}}} \sum_{\substack{t \in T \\ t \neq T}} \sum_{\substack{t' \in T \\ t < t' \leq t + \Delta t}} \frac{\xi_{i,t,t'}}{tph_{r(i)}} \cdot cph_{r(i)}$$

Chapter 4

Implementation

A solution for solving the planning and scheduling problem at AC has been proposed by [4]. Due to the large problem complexity and size, a powerful and generic software tool for planning and scheduling is adapted by means of a custom made plugin. The planning tool used within the process is IBM ILOG Plant PowerOps (PPO) [11]. The advantage of using PPO is that an entire framework is already incorporated for solving generic planning and scheduling problems. Namely, several solving methods and algorithms are already incorporated. Some of these methods are discussed in Chapter 2. Another advantage of using PPO is that a rich GUI is also already included. This is because several convenient views exist to inspect or adapt the produced schedules and also to view their quality. To be able to make PPO usable for the planning and scheduling problem at AC, one has to define an instance of PPO's internally used data model. The initial proposed solution by [4], therefore consists of the following two steps:

1. Define an intermediate model that defines the scheduling problem of AC

2. Write a custom plugin (in the Java language) that maps the intermediate data model to PPO's internally used data model

Since the creation of the plugin and the intermediate data model, several changes have occurred in the organizational context of AC. Several technical constraints, as listed in Section 3.1, have been added and several constraints that were described in [4] have become deprecated and are removed.

Because of the many changes and adaptations, in this chapter both the intermediate data model and the mapping to PPO's internal data model are documented. Furthermore, because the two post processing steps in Chapter 5 work on the level of PPO's internal data model, also a simplified data model of PPO is included. Finally, the implementation of the objectives that correspond to the mathematical model are described.

4.1 Intermediate data model

The intermediate data model is not included in this version of the document as it contains lots of classified information

4.2 PPO's data model

A simplified version of PPO's partially used data model is given in Figure 4.1. Some parts of this diagram are constructed by referencing the documentation of PPO. The classes shaded with a grey background are used by PPO to represent a solution. Hence, regarding the mapping from the intermediate data model to that of PPO, they can be ignored.

The main container of PPO's data model is *IloMSModel* which contains resources and recipes. Resources in PPO have natural numbered capacities and can be shut down at certain times by relating them to a defined calender. Resources can be connected to setup matrices through a so-called *feature*. A setup matrix defines for a given set of states S, the duration of each possible pair $(S \times S) \to \mathbb{N}$.

Recipes are the core data structure of PPO. A recipe in PPO consists of several activities and each activity can be executed in several modes. An activity can have several setup features and a particular state $s \in S$ for each feature. If there exists a mode where a resource is connected to a setup matrix with the same feature as that of the activity, setups are respected between subsequent activities on the same resource. Each mode can use one primary resource and can optionally use several secondary resources. Also a mode in PPO can produce several materials and can be tied to either a fixed or variable processing time. A variable processing time means that the time needed for production is proportional to the amount of material produced. Similar to a resource, a mode in PPO can be connected to a calendar as well. For the materials in PPO a demand can be specified at any given time during the planning horizon.

The solving process of PPO is performed uses the classic decomposition method that is similar to the ones discussed in Section 2.2.3. First a planning problem is solved using MIP. To this end, the time horizon of the schedule is divided into buckets of user preferred sizes. Then the planning engine determines which recipes to produce in what quantity for each time bucket and also allocates them to resources. The latter means that the planning engine already chooses a mode for each planned recipe and each activity. A planned production is an instance of *IloMSProductionOrder* as shown in Figure 4.1. After the planning phase the created production orders, they are batched by the batching engine using a heuristic programming method. The planned production orders are inspected and may be split up into several production orders of the same product respecting the minimum and maximum batch size. Also, for each production order, it is determined which parts of the produced quantity go to the stock or the available customer demands (instances of *IloMSDemand*). Finally, the batched production orders are scheduled, by scheduling all activities belonging to the chosen recipe of each production order. Notice that at this point the scheduling engine can not change the chosen recipes determined by the planning engine, although it may change the chosen mode of some activities. This is a serious drawback as is discussed in Section 5.6. The scheduling engine uses CP for the solving process.

4.3 Mapping

The detailed mapping of the intermediate data model to the PPO model is not included in this version of the document. The mapping involves the creation of several objects in PPO for the elements in the intermediate data model.

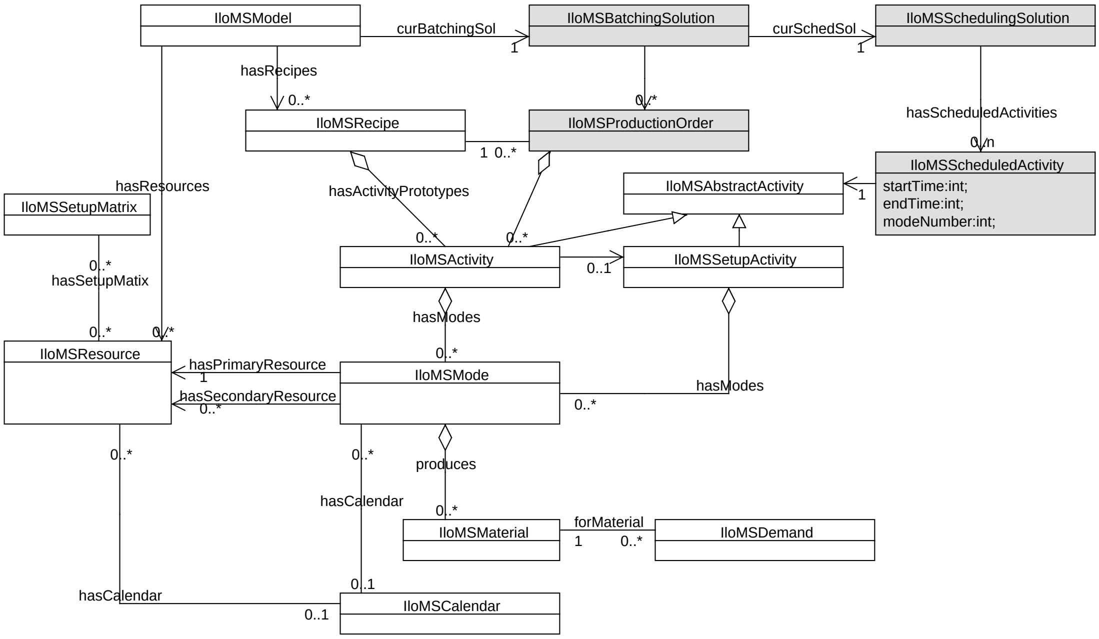


Figure 4.1: UML diagram of PPO's internal data model

4.4 Objectives

Currently, the following objectives are implemented:

4.4.1 Inventory deficit costs

The inventory deficit costs is measured by the total surface that a product quantity is below the stock target of the product. The objective is implemented according to the corresponding formula in the mathematical model. The difference is that in PPO the entire surface below the stock target is seen as inventory deficit costs, instead of only at several time points as in the mathematical model.

4.4.2 Non-delivery costs

These costs are measured according to the formula specified mathematical model. Another implemented objective is the normalized non-delivery cost. These costs are same, except that the multiplier ($ndcModifier_p$) in the mathematical model and *ndcModifier* in the intermediate model are set to 1 for each product. This gives better insights in the actual costs of the produced schedules.

4.4.3 Setup costs

These costs measure the total setup time for all changeovers in the model. This can be multiplied by a multiplier to emphasize importance of fewer changeovers. Less large changeovers, namely spinneret changes, lead to more total production. However it might also be the case that less demands can be satisfied.

Chapter 5

Improvements

In this chapter all the improvements and features added to the original model, in comparison to the version described in [4], are documented. These improvements include a rounding error fix of the planning horizon (Section 5.1), a fix in the original mathematical model regarding the inventory deficit costs objective (Section 5.2), simultaneously optimizing the non delivery and the inventory deficit costs (Section 5.3), and finally, a constraint relaxation regarding the production lines that share a high rise resource (Section 5.4). This is followed by a final comparison of the original model and the model with all implemented improvements in Section 5.5, showing a significant cost reduction. Notice that the improvements, as listed in this chapter, only improve the performance of the initial model. Also, the comparison test results are derived from the same original input data and the same technical constraints as that of the initial model.

5.1 Rounding error

Within the model that is implemented in PPO (see Chapter 4) the time horizon is divided into buckets $b_1, \ldots, b_n$ of a fixed length. Originally, this is done by taking a fixed bucket size w. However, it is almost always the case that H can not be entirely divided by w, for example, if $H = 10$ and $w = 3$. So this means that H falls somewhere before the end of the last bucket (see Figure 5.1). This poses a problem because at time H demands are defined for certain products, which have to be satisfied. The planning engine of PPO, however, does not consider the last bucket (the one H falls into) in the optimization. This leaves several demands at the end of the last month in the planning unsatisfied.

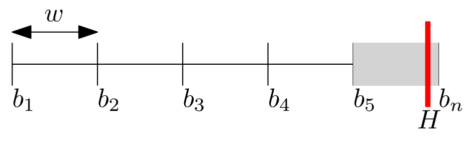


Figure 5.1: Rounding error regarding fixed bucket size and planning horizon H

To fix this rounding error, the plugin now determines the time bucket where H falls into and adjusts this time bucket to fit H. In this manner H falls on the edge of the last bucket and the bucket is now included by the planning engine. A comparison is performed on the test data of *April, May, June 2011* using the default scheduling weights as specified in [4]. This rounding scheme leads to a significant improvement regarding the non delivery costs, however, the inventory deficit costs have increased slightly (see Table 5.1). The latter is likely the case because satisfying demands is more important than satisfying stock targets according to the default planning weights.

	non delivery costs	inventory deficit costs
original	40.66	385.19
rounding fix	26.10	388.27

Table 5.1: Rounding error cost improvement (millions)

5.2 Inventory deficit cost

In the original mathematical model there is an error regarding the inventory deficit costs objective. The inventory deficit costs in [4] are specified as follows:

$$O_{st} = \left(\sum_{p \in P} \left(\sum_{0 < m \leq m_{max}} stc(p) \cdot \frac{0 \uparrow (stockTarget(p) - (0 \uparrow st(p,m))}{stDays(p)} \right) \right) \tag{5.1}$$

p is the set of products, m the month within the planning, $stockTarget(p)$ is the amount in ton of p to be in stock, $st(p,m)$ is the actual stock of p in month m and $stDays(p)$ is the amount of days p should be in stock. Finally, $stc(p)$ specifies the cost per day when p is below the stock target. The objective in equation 5.1 is incorrect because the multiplier $stc(p)$ is specified per day and not per ton. So the amount of days that the stock of p is below the stock target of p should be determined. This can be done correctly by the equation specified as follows.

$$O_{st} = \left(\sum_{p \in P} \left(\sum_{0 < m \leq m_{max}} stc(p) \cdot \frac{(0 \uparrow (stockTarget(p) - st(p,m))) \cdot stDays(p)}{stockTarget(p)} \right) \right) \tag{5.2}$$

Fortunately, in the planning tool the inventory deficit costs were implemented according to formula 5.2, however, in the planning tool the time units are hours and not days, so the total costs should be divided by 24 in the planning tool. This means that all inventory deficit costs of the earlier results in [4] should be divided by 24 as well. This fact does not yield an actual objective improvement, although the relation between the non delivery costs and the inventory deficit costs changes. Thus, for a fair comparison the model weights have to be adjusted regarding the changed inventory deficit costs. Another issue is that in the original model, the inventory deficit costs where also calculated over the extra added time buckets beyond the scheduling horizon. This resulted in inventory deficit costs that were twice the actual size.

5.3 Simultaneous optimization of non-delivery and inventory deficit costs

Within the planning tool, using the original plugin, it is not possible to optimize the non delivery costs and the inventory deficit costs at the same time in the scheduling engine, as mentioned in Section 6.2 of [4]. To cope with this, in [4] a post processing step is proposed that explicitly reduces the inventory deficit costs by fixing the production orders for the non delivery costs. The problem regarding the simultaneous optimization appears to result from a constraint regarding the storage capacities. However, since in the model the storage of materials is omitted, this constraint can be left out. Disabling this constraint allows for simultaneous optimization of the non-delivery and inventory deficit costs and removes the need for the mentioned post processing step in the new model.

5.4 Shunt line constraints

One important requirement in the original model, as mentioned in [4], is the following: The *polymer* concentration for production lines LX and LY can in general maximally differ dx and in some rare cases dy. LY is a shunt line of LX, similar to LM and LN.
In the planning tool using the original plugin, this problem is tackled by defining a setup matrix containing all possible *polymer* concentrations (gathered from all possible production recipes). A fictional example is given in Table 5.2. By using this method the modeling of the *polymer* constraint is too strict, because all pair of recipes that differ in *polymer* can not be produced simultaneously instead of only the ones that differ in *polymer* too much. This drawback comes from the fact that the setup matrix has to be fully filled in the planning tool (each state has to contain a setup to all other states). According to Table 5.2, recipes with *polymer* 0.1 can not be produced together with recipes having *polymer* 0.2 on production lines sharing the same high rise process. However, according to the mathematical model, this should be allowed. To cope with this, in the original model, the possible recipes are copied in the allowed *polymer* range for all shunt lines. In this manner there is always a compatible allowed recipe within the dx range. The scheduling engine however has a lot of trouble selecting the right recipes, requiring a necessary manual post processing step as explained in Section 6.2 of [4].

polymer	0.1	0.2	0.5
0.1		x	x
0.2	x		x
0.5	x	x	

Table 5.2: Example setup matrix, x denotes that two recipes may not be produced simultaneously

The improved plugin uses a different approach instead of the setup matrix. For each high rise resource that is shared by more than one production line, a resource is created for each different *polymer* value . Now the following scheme is applied: One line sharing the high rise resource that produces a recipe seizes all disallowed *polymer* concentrations, while the other production line requires the actual needed *polymer* concentration. In the case of production lines LM LN, this would mean that LM seizes all *polymer* resources that are not compatible with its current recipe, so that LN is forced to produce a compatible recipe (because that does not require a seized resource). The scheme is the same for production lines LX and LY only for a different set of *polymer* resources. Hence the planning engine is forced to plan the right recipes on the shunt lines, as now resources are used to enforce this behavior. A drawback is that this resource scheme only works when at most two production lines share the same high rise process.

The impact of this improvement is quite extensive since there are two pairs of production lines that share the same high rise process. The results regarding the costs reduction using the default scheduling weights (as specified in [4]) are given in Table 5.3.

Performance indicator	Original	*polymer* fix
Non-delivery costs	31.2	10.7
Inventory deficit costs	15.1	16.8

Table 5.3: Influence of the shunt line constraint improvement

5.5 Evaluation

In this section a comparison is performed between the manual planning solution, the original model from [4], the manual post processing steps by [4] and the model containing all improvement steps as described in this chapter so far. The test is conducted on the instance of *April, May, June 2011.* The scheduled downtimes of each resource have been added as well. The scheduling weights for the new model are given in Table 5.4, and for the old model the weights are used as described Section 6 of [4]. Notice, that as explained in Section 5.2, the inventory deficit cost differ by a factor 24 and this is accounted for in the new scheduling weights and also in the comparison between all test instances. Also, the scheduling engine now has a weight set for the inventory deficit costs, because these costs can be simultaneously optimized.

Performance indicator	Planning weight	Scheduling weight
Non-delivery costs	2	2
Inventory deficit costs	1	1
Setup costs	0	0

Table 5.4: Planning and scheduling weights for the improved model

The results in Table 5.5 and Figure 5.2 show a huge improvement with respect to all costs in the improved model. Another important fact is that with the latest model the post processing steps as described in [4] are not required anymore. The setup costs have decreased slightly in the latest model, while not being part of the optimization. However, the setup costs are still much higher than that of the manual planning solution. This confirms that more changeovers can lead to better costs reductions even though time is needed for these changeovers. Notice that the results given in Table 5.5 are based on the old model input data and technical constraints and are only usable for the sake of comparison.

	Non-delivery costs (millions)	Inventory deficit costs (millions)	Setup costs (hours)
Manual	38.90	19.42	309
Original	26.20	21.86	1289
Original post processed	18.60	21.58	1050
Improved	12.01	15.04	956

Table 5.5: Results comparison on test case *April, May, June 2011*

5.6 Automatic post processing steps

Two automatic post processing steps are added to improve the quality of the resulting solutions from PPO. One post processing step reduces the amount of unnecessary spinneret

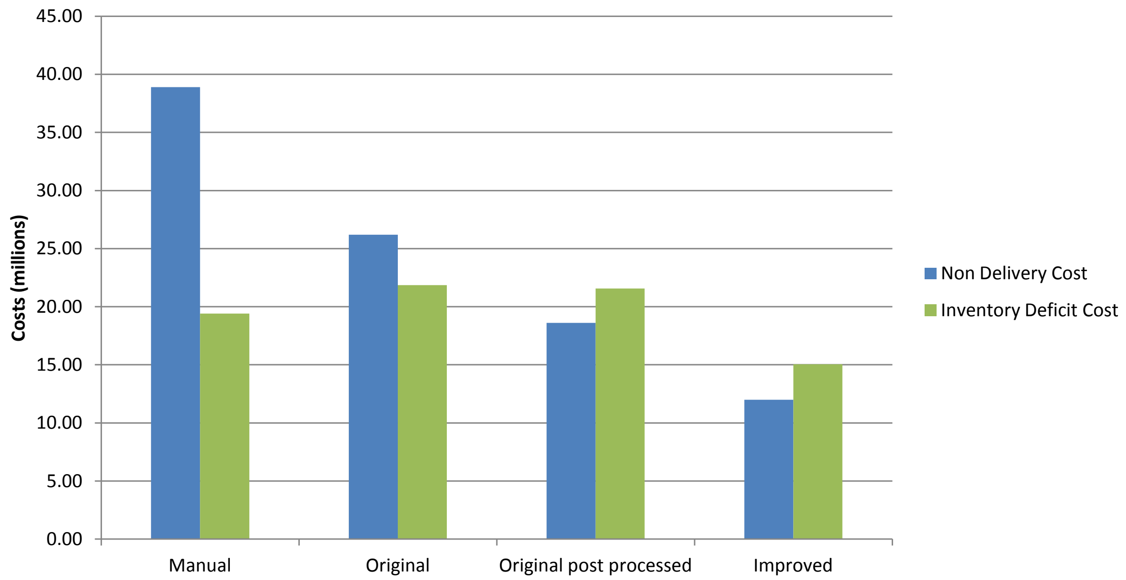


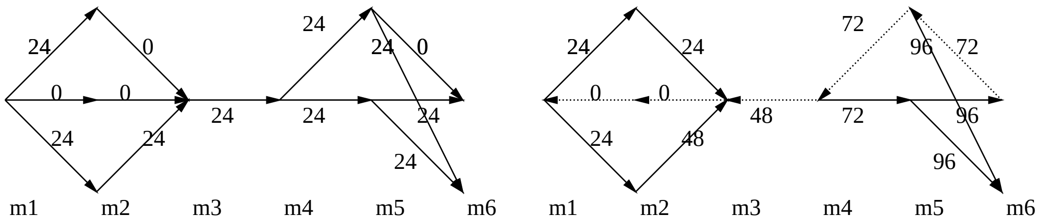
Figure 5.2: Results comparison on test case *April, May, June 2011*

changeovers (Section 5.6.1). Because in PPO a planning problem is solved first, where it is decided what products are produced in which time buckets and also what recipes are to be used, the scheduling engine can not change these recipes. For the AC scheduling problem this leads to quite a substantial number of unnecessary changeovers per production line. The second post processing step inflates the scheduled production orders (Section 5.6.2). The latter is necessary because the production lines contain many small gaps between the scheduled production orders of PPO. These gaps can often be filled with useful productions. The impact of the post processing steps is evaluated in 5.6.3.

5.6.1 Minimizing changeovers

This post processing steps scans for each production line (i.e. the corresponding instances of IloMSresource), the products to be produced and finds possible alternative recipes for these products actively minimizing the amount of changeovers. This is illustrated in Figure 5.3(a). The nodes in the graph represent the possible recipes for each material ($m1 \dots m6$) belonging to the production orders. The edges represent the changeover cost to each subsequent node. The shortest path in this directed graph represents a set of recipe choices (nodes along the path) that lead to a minimum amount of scheduled changeovers. To find the shortest path for this special type of graph, a dynamic programming algorithm has been implemented. The implementation of the algorithm is shown in Figure 5.3(b). After the possible alternative recipes are gathered for each material, the algorithm calculates the total minimum setup required for each material. The edges in Figure 5.3(b) represent the total minimal setup costs so far. This starts from the leftmost material and ends at the rightmost material. Next, working backwards, the minimal path is chosen and the production orders are adapted accordingly.

After the shortest path is found the scheduled production orders on the production line are assigned the newly chosen recipes. This might result in changeovers that are removed and in some cases a changeover might be introduced. The overall total number of changeovers is minimal however. In the worst case scenario an introduced changeover might be scheduled during the weekend, as this is difficult to detect by the algorithm. Should this occur, slight manual post processing has to be done to rearrange the allocated changeovers.



(a) Post processing step 1 (b) Post processing step 2

Figure 5.3: Post processing step for a single production line

The removed changeovers lead to gaps that can possible be filled by the subsequent post processing step as discussed in the next section. Should filling not be possible (due to for example a resource constraint) a little manual post processing can be used to fix the small gap.

5.6.2 Inflating the scheduled production orders

This post processing step inflates the scheduled production orders as much as possible, until either a resource conflict arises or a scheduled downtime is reached. The procedure works entirely in the data model of PPO and can be applied to production orders of any generic PPO problem. The pseudocode of the post processing step is given in Algorithm 1.

Algorithm 1 InflateProductionOrders

---

for *po* instance of *IloMSProductionOrder* **do**
  minGap ← scheduling horizon
  for instances of *IloMSScheduledActivity ac* belonging to *po* **do**
    mG ← scheduling horizon
    for instances of *IloSMResource r* in scheduled mode of *ac* **do**
      mG ← min(*mG*, size *ac* can increase without exceeding capacity of *r*)
      if *r* has an upcoming downtime after *ac* **then**
        mG ← min(*mG*, next downtime of *r*)
      end if
    end for
    $minGap \leftarrow \min(minGap, mG)$
  end for
  increase production order size to found *minGap*
end for

5.6.3 Evaluation

To determine the impact of both post processing steps, the instance of *April, May, June 2011* is evaluated. After PPO has found a scheduling solution, the results are compared with and without the post processing steps applied.

The results given in Table 5.6 indicate that inflating the production orders reduces the costs slightly. However, the inventory deficit cost have increased a little. This can be explained by an introduced spinneret changeover that removes a day production from a certain production line. Of course more other changeovers are removed, but they might

Performance indicator	Before	After
Non-delivery costs (million)	17.5	17.3
Normalized non-delivery costs (million)	10.4	10.3
Inventory deficit costs (million)	6.7	6.8
Setup time (hours)	1057	758
Spinneret changes	34	19

Table 5.6: Performance impact post processing *April, May, June 2011*

contribute less (or not at all) towards meeting important stock targets. The removal of a unnecessary spinneret changeover is however always better cost wise.

Chapter 6

Evaluation

In order to verify and validate the quality of the final model after all model improvements (see Chapter 5), an evaluation study has been conducted. In order to verify that the output of the final model is correct and satisfies the technical constraints, a comparison is made between two manual planning solutions and the corresponding schedules by the model (Section 6.1). Also, to verify that the model generates solutions that are correct and intuitive, some validation experiments have been conducted in 6.2. Finally, several what-if scenarios and experiments are evaluated in Section 6.3.

6.1 Verification manual planning

In order to compare the model schedules and the manual schedules, two manually designed schedules are created in the model and compared to the output of the model on the same input data. The default objective weights used for this verification are given in Table 6.1 and correspond to the chosen weights in [4], having the inventory deficit costs weight corrected (see Section 5.2 of Chapter 5). The time used for the planning engine was 500 seconds and the scheduling engine is executed till completion (after about 2500 seconds). The planning engine does not find substantially better solutions after 200 seconds and is therefore capped to 500 seconds.

Performance indicator	Planning weight	Scheduling weight
Non-delivery costs	2	2
Inventory deficit costs	1	1
Setup costs	0	0

Table 6.1: Planning and scheduling weights used for the comparison between the manual and model schedules

6.1.1 Instance *April, May, June, 2011*

During the creation of the manual schedule in the model, several technical constraints are violated. The reason is that some of these constraints were not (yet) taken into account during the planning process. Furthermore, some constraints turn out to be not so strict in practise. To make a fare comparison between the model and the manually created schedules some technical constraints have been relieved.

Also, the scheduled downtimes and fixed productions present in the manual schedule are added to the schedule of the model. The summarized results are shown in Table 6.2. There is a large reduction in the top three costs objectives for the model output. The amount of spinneret changes, setup time, and the total produced amount is worse than that of the manual planning though. It should be mentioned some an important factor was not accounted for in the model.

In the month of April, a very large production order for the product *2040930_A* is satisfied by the model, however a lot less is satisfied by the manual schedule. This is because the forecast of that product has been reduced later and this is adapted afterwards in the manual schedule, but not in the forecast data of that moment. The model hence uses outdated forecast data. The model result is thus probably of lesser quality, than is shown in Table 6.2 regarding the non-delivery costs objective.

Performance indicator	Manual planning	Model output
Non-delivery costs (million)	28.9	17.0
Normalized non-delivery costs (million)	16.0	10.3
Inventory deficit costs (million)	11.5	6.8
Setup time (hours)	350	697
Spinneret changes	4	19
Total produced (ton)	5241	4847

Table 6.2: Comparison results *April, May, June 2011*

6.1.2 Instance *Oct, Nov, Dec 2011*

Here some constraints are violated as well, during the creation of the manual schedule in the model.

Also, the scheduled downtimes and fixed productions present in the manual schedule are added to the schedule of the model. Another important remark is that within this comparison no strategically important product have been emphasized. This means that the non-delivery costs and normalized non-delivery cost are equal.

Performance indicator	Manual planning	Model output
Non-delivery costs (million)	5.3	0.1
Inventory deficit costs (million)	12.5	7.3
Setup time (hours)	350	975
Spinneret changes	8	29
Total produced (tons)	4842	4574

Table 6.3: Comparison results *Oct, Nov, Dec 2011*

The summarized results are shown in Table 6.3. Interestingly, the non-delivery costs are closer to each other in comparison to the results of *April, May, June, 2011.* This may result from the fact that there are no forecast data errors like there were in the instance of *April, May, June, 2011.* Although the model shows a cost reduction, many more setup time is used and especially a lot more spinneret changeovers. This large amount of changeovers is even present after minimizing the amount of setups per production line by the first post processing step, as discussed in Section 5.6.1. The reason for this might be the extra constraints that have been added compared to those present in the instance of *April, May, June 2011*.

Notice that also in this case, the model schedules still shows some small gaps. These gaps are mostly at the start of the production lines, because a minimum of 5 days production is required in the model and the gaps are too short to fill with production. In practice, however, this could be resolved by lengthening the previous running recipes for one or two days.

6.2 Validation

Some validation experiments are conducted to improve the confidence in the model. The results from these experiments are considered as classified information and have been omitted in this version of the document.

6.3 Analysis

6.3.1 Nondeterminism

A problem with PPO is that it suffers from nondeterminism. This means that different runs (i.e., solving the same instance multiple times), can lead to different results. To investigate this behavior, a similar problem instance of *April, May, June 2011* is solved multiple times and the cost outcome is summarized. Because the outcome of each run is independent, a 95% confidence interval can be constructed using the normal distribution. This only holds if we have sufficient runs.

Run	Non-delivery	Non-delivery (n.)	Inventory deficit	Setup
1	12.99	8.49	6.84	745
2	13.34	8.85	6.87	674
3	12.09	8.19	7.41	689
4	12.46	7.90	10.76	859
5	12.34	7.99	10.84	693
6	10.19	7.25	9.02	811
7	13.30	8.80	6.88	683
8	12.11	7.77	10.86	690
9	13.30	8.80	6.90	690
10	13.30	8.80	6.89	683
11	10.55	8.03	8.77	614
12	12.42	7.89	7.21	786
13	11.94	7.55	7.54	675
14	11.50	8.07	7.06	753
15	12.08	8.32	7.00	785
Min	10.19	7.25	6.84	614
Max	13.34	8.85	10.86	859
Avg.	12.26	8.18	8.06	722
Std. Dev.	0.96	0.49	1.58	65.16
95% conf.	[11.77;12.75]	[7.93;8.43]	[7.26;8.85]	[689.02;754.98]

Table 6.4: Costs summary of multiple runs of the instance of *April, May, June 2011*

The results (Table 6.4) show a cost summary for the (normalized) non-delivery costs and inventory deficit costs (million euros), and also the setup costs (in hours). This overview shows that there exists a large variation in the outcome of the same problem instance. This means that when comparing the model solutions of two problem instances, no reliable conclusions can be drawn if only a single run is executed (the outcome of one problem instance, might be better by coincidence). To fairly evaluate the costs of a single scenario, the average of multiple runs has to be used. The number of runs should be sufficiently large as well. Notice that for even 15 runs, the confidence intervals are still fairly large. In order to make reliable predictions when comparing the outcome of the model, at least 20 to 30 runs will probably have to be used.

6.3.2 Non-delivery versus inventory deficit costs

One of the improvements, as stated in Chapter 5, is that the non-delivery and inventory deficit costs can now be simultaneously optimized. Therefore an analysis has been made to compare the weights used for these costs. To this end the instance of *April, May, June 2011* is used. Notice that the since we are now comparing between model results, the scheduled downtimes and fixed productions are not added to the schedules of the model. Note that it is not useful to investigate scenario's with higher weights for the inventory deficit costs, since at AC it is always a higher priority to satisfy demands over stock. Furthermore, only one instance is solved for each weight setting. The results of the nondeterminism investigation in Section 6.3.1 show that different runs should could be used for each weight setting, to be more accurate. However, the purpose of this experiment is to investigate the impact of the weights rather than giving a true cost prediction.

Weights		**Objectives**	
Non-delivery	Inventory deficit	Non-delivery	Inventory deficit
1	0	7.5	12.9
4	1	11.7	6.4
2	1	11.1	7.4
4	3	11.9	6.7
1	1	10.3	7.3

Table 6.5: Comparison results *Oct, Nov, Dec 2011*

The results in Table 6.5 show that there is little variation in the total non-delivery costs, regardless of the used weights. This does no hold if the weight is zero for the inventory deficit costs. In that case, the non-delivery costs are much lower compared to the other test results. The weight distribution does however seem affect the inventory deficit costs by a large amount. The best results are achieved when the weight of the non-delivery costs is set not too much higher than that of the inventory deficit costs. Apparently, putting to much weight on the non-delivery costs, results in the model putting too much effort in satisfying few demands, while it is easier to satisfy the stock targets.

6.3.3 Non-delivery versus setup costs

An analysis has been performed to investigate the influence of these costs on the model, because the costs regarding changeovers of type *spinneret* are very large for AC. To this end the setup costs of spinneret changeovers are gradually increased and the impact is observed on the non-delivery costs. The model is adjusted such that only for spinneret changeovers costs are introduced. The costs of a spinneret changeover is derived in the same way as was done in [4] by taking the average daily contribution per production line of 50000 euros per day.

Weights		**Objectives**	
Non-delivery	Setup cost	Non-delivery	nr. of spinneret changes
1	0	11.0	15
1	1.5	9.4	14
1	1	13.1	11
1	2	9.6	11
1	3	14.2	6
1	4	8.9	11
1	6	13.2	7
1	8	12.1	6
1	10	12.4	7
1	15	13.2	10
1	20	17.3	10
1	25	26.1	4

Table 6.6: Non-delivery versus setup costs

The results from Table 6.6 indeed show that when the setup costs are gradually increased, the number of spinneret changes are generally reduced in the schedules generated by the model. This comes at the costs of the non-delivery costs. Notice that these results are still very susceptible to PPO's non-determinism. However, even with the first post processing step enabled (Section 5.6.1), the model is unable to find the minimal number of setups that can be used as the setup costs get very large. Namely, the manual solution shown in Section 6.1, shows that a solution exists that uses just four spinneret changeovers. The best solution found by the model (regarding the setups) also uses four changeovers and has comparable costs, however, visually inspecting the model solution shows many large gaps and even an empty production line. From this we can conclude that the PPO model still has issues regarding the setup minimization. This result weakens the initial assumption that more changeovers are the necessary result of a better cost reductions. However, note that this result does not invalidate the assumption that minimizing the amount of setups does not minimize the costs.

Chapter 7

Conclusion

The goal of this project was to improve the existing planning and scheduling solution for the production process at AC. The initial proposed solution by [4] showed potential and several improvements have been identified since then. One type of improvements involve adaptations made to the data model used by the planning tool. This is achieved by making adaptations to the mapping from the intermediate data model that is used to contain a problem instance. To compare the performance of the current model with the original one by [4], a comparison was made between a manually created schedule and the schedule outcomes of both models. The following results have been found:

- The two manual post processing steps needed to improve the quality of the produced schedules of the original model are not required for the new model.
- The non-delivery costs and inventory deficit costs showed a cost reduction of 60% and 20% respectively in the current model. This is against the 17% and even an increase of inventory deficit costs for the first model. Moreover the setup costs used by the current model are reduced slightly although not part of the optimization. Note that these results were obtained using the original input data.
- The resulting output of the schedules lead to less gaps and better filling of the heavily constrained production lines.

To further boost the model's performance an automatic post processing step is implemented that inflates the production orders of the produced schedules and removes the few small gaps that still existed (if possible). Although this only leads to minor cost reductions, the resulting schedules look more realistic. The new model has been thoroughly tested and the output is reviewed in detail. As a result, some technical constraints have been changed and refined in the current model. Also, several large errors in the input data have been corrected. These adaptations greatly influenced the models performance and also increased the quality of the manual plans when put into the model. In order to evaluate the current modeling solution better, a new objective is added to measure the normalized non-delivery costs.

To evaluate the current model a new comparison is made for two problem instances between the model outcomes and the corresponding manual schedules. The model output of *April, May, June, 2011* shows a reduction of 41% in non-delivery costs and another 41% in inventory deficit costs. However, a large portion of the difference in non-delivery costs has been explained by outdated forecast data in the models input data. The model results of *Oct, Nov, Dec 2011* show a schedule with almost no non-delivery costs at all compared to the manual plan and also a reduction of inventory deficit costs by 42%. This leads to an overall costs difference (non-delivery plus inventory deficit) of 60%. From this perspective the model quality is good, however, the setup cost objective suggests there may be room for improvement. Also, be aware that not all cost objectives have been implemented yet.

The schedules produced by the model contain more large changeovers (of type spinneret). As suspected by [4], this difference was due to the many more different demands being satisfied by the model. However, after more careful inspection, the scheduling engine tends to generate unnecessary changeovers. Even though another post processing step has been implemented to remove all unnecessary changeovers per production line, the scheduling engine still has issues when trying to minimize the amount of setups used. This has been confirmed with an experiment in Section 6.3.

Although the current model might be suboptimal regarding the setup time reduction, it is still of better quality than the manual schedules, i.e., leads to a better objective function value. Also the model can still be used to evaluate several interesting what-if scenarios. Furthermore, as part of the testing, and in order to increase the confidence in the current model, several validation experiments have been performed. To be able to use the current planning solution for the comparison between several what-if scenarios however, one has to be aware of the nondeterministic behavior regarding PPO. This behavior has been investigated with another experiment in Section 6.3.1.

7.1 Future work

As stated in the conclusion, the performance of the current model has thoroughly been improved. Also, the model is better comparable to the manual planning. However, there is still an open issue regarding the minimization of the amount of changeovers. Furthermore, due to time constraints and many last minute changes to the model (input data and constraints), several experiments have not yet been performed. Hence the following suggestions are made for future work:

- The current planning and scheduling engine of PPO is suboptimal regarding the minimization of setup costs. An improvement would be to write a custom engine that can handle the setup costs better. Given the environment of PPO the best option would be to use CP for this. MILP can also be used, however, some constraints and objective functions might be difficult to linearize.
- Investigate the use of a totally different scheduling solution (outside of PPO). Several possibilities have been identified in the literature (Chapter 2). This could be used as a second verification method along side of the current PPO solution. Optionally, this solution can implement the MILP formulation as discussed in Chapter 3. An advantage of using MILP for the entire solving process is that there exist many deterministic solving algorithms, so that the current nondeterminism problem of PPO would not be an issue.
- Implement the remaining optimization criteria regarding the transportation costs and also resort allocation.
- Continue to investigate scenarios regarding the configuration of production lines and technical constraints. To this end the current solution can be used as a tool. It has been shown that several runs of each scenario have to be compared though, in order to make reliable predictions.

Nomenclature

High rise resources

$hrCap_h$	throughput capacity of h in kilos per hour

Production lines

$hrRes_l$	high rise resource h used by production line l
$minTp_l$	minimum throughput required on l
$name_l$	identification of l

Model parameters

Δt	maximal number of events points a task can overlap
dem_o	specifies the quantity of production order o
due_o	due time of production order o
$E_e^{\max}$	maximum capacity of resource e
cht_i	changeover time (in hours) needed for changeover task i
$\mu_{i,e}$	required capacity of resource e by task i
$ndcModifier_o$	modifier to enforce strategically important value of production order o
pr_o	specifies the product of production order o
$l(i)$	production line l of task i
$r(i)$	recipe r of task i
H	time horizon
I_p^0	initial stock of product p

Objectives

O_{ndc}	total non-delivery costs
O_{pc}	total production costs
OF	total objective outcome
w_{ndc}	weight of non-delivery costs
w_{ndc}	weight of stock deficit costs
w_{pc}	weight of production costs

Products

$profit_p$	profit of product p per ton in euros
stc_p	cost *per day* of supply when the stock is below the stock target
$stDays_p$	total days of sales p corresponding to stock target
$stockTarget_p$	stock target of product p in tons

Recipes

$allowSim_{r,r'}$	wether if two recipes r, r' are allowed to be produced concurrently
mpt_r	minimal processing time (in hours) of recipe r
pr_r	product p produced by recipe r
$spinneret_r$	required spinneret for r
$spoolweight_r$	spool weight of r
$titer_r$	weight (in grams per 10 kilometers) of the product produced by r
tp_r	throughput required of the high-rise resource for r in kilos per hour
tph_r	the number of tons produced per hour by r
ttf_r	type, titer and filaments identification of the product produced by r
$type_r$	integer type number of the product produced by r (without the optional "D" in front of it)

Sets/indices

I^{co}	changeover tasks
I^{spin}	spinning tasks
E, e	generalized model resources
H, h, h'	high rise resources
I, i, i'	processing tasks
L, l	production lines
O, o, o'	production orders
P, p, p'	products
R, r, r'	production recipes
S, s	spinneret types
T, t, t'	event points

Variables

$po_{o,t}$	binary variable deciding wether production order o is (partially) fulfilled at event point t
$psl_{o,t}$	slack (unfulfilled demand) of production order o at event point t
$\xi_{i,t,t'}$	total amount of material processed by task i, starting at t and finishing at t'
$E_{e,t}$	excess amount of resource e at event point t
$I_{p,t}$	amount of product p at event point t

$N_{i,t,t'}$	binary variable that assigns the end of task i, which began at t, to point t'
T_t	absolute time of event point t

Bibliography

[1] Bassett, M. H., Pekny, J. F., and Reklaitis, G. V. Decomposition techniques for the solution of large-scale scheduling problems. *AIChE Journal 42*, 12 (1996), 3373–3387.

[2] Bowman, E. H. The schedule-sequencing problem. *Operations Research 7*, 5 (1959), pp. 621–624.

[3] Castro, P. M., Barbosa-Pvoa, A. P., Matos, H. A., and Novais, A. Q. Simple continuous-time formulation for short-term scheduling of batch and continuous processes. *Industrial & Engineering Chemistry Research 43*, 1 (2004), 105–118.

[4] Coset, R. Modeling and solving the production scheduling problem at ac. Master's thesis, Technische Universiteit Eindhoven, 2011.

[5] Dogan, M. E., and Grossmann, I. E. A decomposition method for the simultaneous planning and scheduling of single-stage continuous multiproduct plants. *Industrial & Engineering Chemistry Research 45*, 1 (2006), 299–315.

[6] Erdirik-Dogan, M., and Grossmann, I. E. Simultaneous planning and scheduling of single-stage multi-product continuous plants with parallel lines. *Computers & Chemical Engineering 32*, 11 (2008), 2664 – 2683. Enterprise-Wide Optimization.

[7] Floudas, C., and Lin, X. Mixed Integer Linear Programming in Process Scheduling: Modeling, algorithms, and applications. *Annals of Operations Research 139* (2005), 131–162. 10.1007/s10479-005-3446-x.

[8] Floudas, C. A., and Lin, X. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Computers & Chemical Engineering* (2004), 2109–2129.

[9] Harjunkoski, I., and Grossmann, I. E. Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods. *Comp. Chem. Engng 26* (2002), 1533–1552.

[10] Hooker, J. N. An integrated method for planning and scheduling to minimize tardiness, 2006.

[11] IBM. Information about IBM ILOG Plant PowerOps. `http://www.ibm.com/software/integration/sca/plant-powerops/about/`.

[12] Janak, S. L., Floudas, C. A., Kallrath, J., and Vormbrock, N. Production scheduling of a large-scale industrial batch plant. i. short-term and medium-term scheduling. *Industrial & Engineering Chemistry Research 45*, 25 (2006), 8234–8252.

[13] Kallrath, J. Planning and scheduling in the process industry. *OR Spectrum 24* (2002), 219–250. 10.1007/s00291-002-0101-7.

[14] Kondili, E., Pantelides, C., and Sargent, R. A general algorithm for short-term scheduling of batch operations. *Computers & Chemical Engineering 17*, 2 (1993), 211 – 227. An International Journal of Computer Applications in Chemical Enginnering.

[15] Li, W. D., and McMahon, C. A. A simulated annealing-based optimization approach for integrated process planning and scheduling. *Int. J. Comput. Integr. Manuf. 20* (January 2007), 80–95.

[16] Li, Z., and Ierapetritou, M. Process scheduling under uncertainty: Review and challenges. *Computers & Chemical Engineering 32*, 4-5 (2008), 715 – 727. Festschrift devoted to Rex Reklaitis on his 65th Birthday.

[17] Li, Z., and Ierapetritou, M. G. Integrated production planning and scheduling using a decomposition framework. *Chemical Engineering Science 64*, 16 (2009), 3585–3597.

[18] Lin, X., Floudas, C. A., Modi, S., and Juhasz, N. M. Continuous-time optimization approach for medium-range production scheduling of a multiproduct batch plant. *Industrial and Engineering Chemistry Research 41* (2002), 3884–3906.

[19] Liu, Y., and Karimi, I. Scheduling multistage, multiproduct batch plants with nonidentical parallel units and unlimited intermediate storage. *Chemical Engineering Science 62*, 6 (2007), 1549 – 1566.

[20] Maravelias, C. T., and Sung, C. Integration of production planning and scheduling: Overview, challenges and opportunities. *Computers & Chemical Engineering 33*, 12 (2009), 1919 – 1930. FOCAPO 2008 Selected Papers from the Fifth International Conference on Foundations of Computer-Aided Process Operations.

[21] Mendez, C. A., Cerda, J., Grossmann, I. E., Harjunkoski, I., and Fahl, M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & Chemical Engineering 30*, 6-7 (2006), 913 – 946.

[22] Omar, M. K., and Teo, S. C. Hierarchical production planning and scheduling in a multi-product, batch process environment. *International Journal of Production Research 45*, 5 (2007), 1029–1047.

[23] Pantelides, C. Unified frameworks for optimal process planning and scheduling, 1994.

[24] QIAN, Y., PAN, M., and HUANG, Y. Modeling and optimization for scheduling of chemical batch processes. *Chinese Journal of Chemical Engineering 17*, 1 (2009), 1 – 7.

[25] Sundaramoorthy, A., and Karimi, I. A simpler better slot-based continuous-time formulation for short-term scheduling in multipurpose batch plants. *Chemical Engineering Science 60*, 10 (2005), 2679 – 2702.

[26] Sundaramoorthy, A., and Maravelias, C. T. Simultaneous batching and scheduling in multistage multiproduct processes. *Industrial & Engineering Chemistry Research 47*, 5 (2008), 1546–1555.

[27] Wassick, J. M., and Ferrio, J. Extending the resource task network for industrial applications. *Computers & Chemical Engineering 35*, 10 (2011), 2124 – 2140.

[28] Wu, D., and Ierapetritou, M. Hierarchical approach for production planning and scheduling under uncertainty, 2007.

[29] Zeballos, L. J., Novas, J. M., and Henning, G. P. A CP formulation for scheduling multiproduct multistage batch plants. *Computers & Chemical Engineering 35*, 12 (2011), 2973 – 2989.

[30] Zhang, H., and Gen, M. Effective genetic approach for optimizing advanced planning and scheduling in flexible manufacturing system, 2006.

[31] Zhang, X., and Sargent, R. The optimal operation of mixed production facilities: a general formulation and some approaches for the solution. *Computers & Chemical Engineering 20*, 6-7 (1996), 897 – 904. Fifth International Symposium on Process Systems Engineering.